Run by Run Process Control

by

Ármann Ingólfsson

B.S. in Industrial Engineering, University at Buffalo (1989)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1991

Author .... [illegible] ....
Department of Electrical Engineering and Computer Science
August 30th, 1991

Certified by .... [illegible] ....
Emanuel Sachs
Associate Professor of Mechanical Engineering
Thesis Supervisor

Accepted by .... [illegible] ....
Thomas L. Magnanti
Codirector, Operations Research Center

Run by Run Process Control

by

Ármann Ingólfsson

Submitted to the Department of Electrical Engineering and Computer Science
on August 30th, 1991, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

A process control algorithm, referred to as the Run by Run (RbR) Controller, is developed and its performance is analyzed in this thesis. The RbR controller is designed to maintain the performance of a production process that is subject to unpredictable disturbances. This is done by regulating the process using feedback control and at the same time continuously diagnosing the state of the process using a generalized version of Statistical Process Control (SPC).

The three major components of the RbR controller are the gradual, rapid, and diagnostic modes. The gradual mode is the default mode, in which a feedback control law based on the EWMA statistic compensates for slow but steady drift in the process mean. The diagnostic mode employs Generalized SPC to monitor the process for the presence of large shifts in the process mean. When the diagnostic mode signals an alarm, rapid mode is entered, wherein an attempt is made to recover the performance of the process before the shift as quickly as possible. In both gradual and rapid modes, the control strategy consists of using measurement data to continuously update a first-order process model, and then selecting a process recipe for which the output is expected to be on target. The first-order coefficients of the model are fixed, whereas the intercept term is updated. Extensions to second-order models are discussed.

The stability and robustness of the gradual mode is analyzed under various assumptions about the behavior of the process to be controlled. Simple conditions that guarantee stability are derived. These conditions relate the estimated process gain to the true process gain, and the EWMA weighing scheme. An expression for the expected variation of the process output as a function of the true process parameters, the estimates of these parameters, and the EWMA weighing scheme are derived. Also, the performance of a complete implementation of the RbR controller in controlling a simulated second-order process was evaluated using a designed experiment. The results show that the RbR controller is robust against modest departures of the process behavior from the idealized first-order model used by the algorithm.

Thesis Supervisor: Emanuel Sachs
Title: Associate Professor of Mechanical Engineering

Acknowledgments

Til foreldra minna.

I would like to thank my thesis supervisor, Emanuel Sachs, for his constant encouragement and valuable insight. When I started working on this project, I expected it to be both challenging and rewarding, and I have not changed my mind in that respect. The other students working on the Process Optimization and Control project: Andy Ruey-Shan Guo, Sungdo Ha, Albert Hu, Jay Ochoco, and Stefan Thomke; and Bernd Mittmann all deserve my thanks. Had I not been able to discuss ideas with them, the quality of this thesis would have been considerably less.

I also wish to thank IBM for supporting this research, and special thanks are due to Bob McCafferty, whose feedback and support were very valuable.

I thank the staff and students at the Operations Research Center for providing me with a supportive environment. I will not mention any names, because I cannot think of any that I would want to leave out.

Finally, I thank my parents and family, for their love and for always having been supportive of my decisions.

Contents

1 Introduction **6**

1.1 Goal of the Thesis 8

1.2 Literature Review 12

1.2.1 Statistical Process Control 12

1.2.2 Automatic Process Control 15

1.2.3 Paper by MacGregor ([Mac87]) 23

1.2.4 Paper by Box and Kramer ([BK90]) 23

1.2.5 Algorithmic Statistical Process Control 25

1.2.6 Papers by Baxley ([Bax91], [Bax90]) 25

1.2.7 Taguchi's "Beta coefficient" Method 26

1.3 Remainder of the Thesis 27

2 Setting **29**

2.1 Role of the RbR Controller 31

2.2 VLSI Fabrication 33

2.2.1 Modeling Assumptions 34

3 Overview of the Algorithm **39**

3.1 Generalized SPC 40

3.2 Process Model 44

4 Gradual Mode **46**
4.1 Estimating the Process Level 46
4.1.1 Bayesian updating 52
4.2 Selecting New Recipe 57
4.2.1 Second-order Process Model 59

5 Rapid Mode **64**
5.1 Estimation of Step Location and Magnitude 65
5.1.1 Formal Description of the Estimation Procedure 66
5.2 Assessing the Certainty of Shift 68
5.2.1 The Bayesian Updating Procedure 69
5.2.2 The Current Algorithm 72
5.3 Enhancements 74

6 Stability and Robustness **76**
6.1 Motivation 76
6.2 First Order Deterministic Process 79
6.3 Deterministic Second Order Process 84
6.4 Drifting First Order Process 88
6.4.1 Deterministic Drift 91
6.4.2 Random Drift 94

7 Testing the Algorithm **99**
7.1 Implementation of the Algorithm 99
7.2 Experimental Design 100
7.2.1 Outer Array 101
7.2.2 Inner Array 103
7.3 Results 104

8 Conclusions **112**

Bibliography **115**

Chapter 1

Introduction

Process control is an important part of the operation of any manufacturing process. The goal of process control may be broadly stated as maintaining the performance of a process at the best possible level over time.

To formalize this notion, suppose that $\mathbf{x}$ is a vector of input variables to the process, that can be observed and possibly adjusted, and $\mathbf{y}$ is a vector of response variables. The input variables could be dial settings, which can be adjusted, or environmental conditions, such as temperature, which can be monitored but not adjusted. The response variables could be a dimension, resistivity, a growth rate, or any other measure used to judge the quality of the product and, by inference, the performance of the process. Mathematically, the process can be represented as a time dependent function f_t which maps a given value of $\mathbf{x}$ into $\mathbf{y}$:

$$\mathbf{y}_t = f_t(\mathbf{x}_t)$$

Even though it is not shown explicitly, $\mathbf{y}_t$ is a function not only of the input variables $\mathbf{x}_t$, but also of some sample space. In other words, $\mathbf{y}_t$ is a random variable. The uncertainty stems from such sources as measurement errors, inaccurate dial settings, or the effects of unkown or ignored input variables.

The performance criteria, z_t, is some function of the response vector, i.e. $z_t = g(\mathbf{y}_t)$. It is useful to distinguish between the activities of *optimization* and *control*, both of which will be considered process control activities. A set of values of the input variables is referred to as the current *recipe*. Optimization consists of finding the recipe

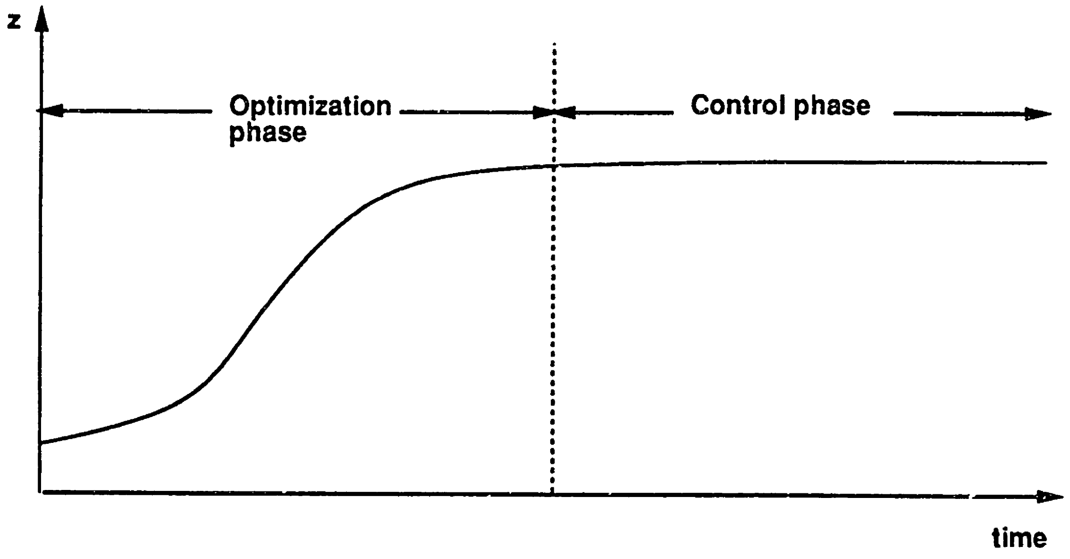


Figure 1-1: The concepts of optimization and control illustrated

$\mathbf{x}_t$ that result in the best value of z_t. This activity may be repeated periodically, if it is thought that additional possibilities for improvement exist, or if the function f_t has changed drastically. Control is concerned with keeping the performance of the process at the optimal level. Figure 1-1 illustrates these concepts. With reference to this figure, the goal of process control can be restated as maximizing the area under the performance criteria curve (if the criteria is to be maximized). In this thesis, performance will be measured primarily by the mean squared deviation of the output from target.

A primary concern when performing process control activities is striking the proper balance between exploring different recipes in the search for better performance, and the cost of making product that does not meet specifications (scrap). Approaches to process control can be conveniently classified as follows according to how this trade-off is made:

On-line process control methods are used during regular production, i.e. the amount of scrap is kept to a minimum.

Off-line process control is at the other extreme, where the cost of scrap is ignored.

The algorithm presented in this thesis is an on-line process control procedure, whose function is to maintain the performance of a process, as opposed to improving it.

1.1 Goal of the Thesis

With this broad framework in mind, the goal and scope of this thesis can be described. One step of a batch manufacturing process will be considered (this step will be referred to as the *process*). A *run* will refer to the processing of one batch. The runs will be identified with a discrete time index $t = 1, 2 \ldots$. Before run t, values for the controllable input variables $\mathbf{x}_t$ are selected, and during or after run t the values of the output variables $\mathbf{y}_t$ are measured.

The goal is to develop an on-line process control strategy that reduces the variability of the output by effective regulation, while at the same time allowing the process to be continuously monitored to search for assignable causes of variation. The regulation procedure will at any given time be in one of two modes: *gradual mode* or *rapid mode*. Gradual mode is the default mode and its purpose is to compensate for *drift* in the process. Rapid mode is entered when there is significant evidence that a *shift* has occurred. A high-level flowchart of the proposed algorithm is shown in figure 1-2. It is evident from the flowchart that the algorithm has three major components:

A diagnostic mode , which decides whether the process is behaving in accordance with the current process model. This mode may be thought of as a generalized version of Statistical Process Control.

A gradual mode , that gradually modifies the process model.

A rapid mode , which quickly updates the process model.

Most of this thesis will be devoted to presenting the steps of the diagnostic, gradual, and rapid modes of the RbR controller, and to evaluate its performance, both analytically and by simulation.

Justification for Gradual and Rapid Modes

The control mode of the RbR controller is designed to accommodate the types of situations that we expect will be most commonly encountered. Two generic situations are:

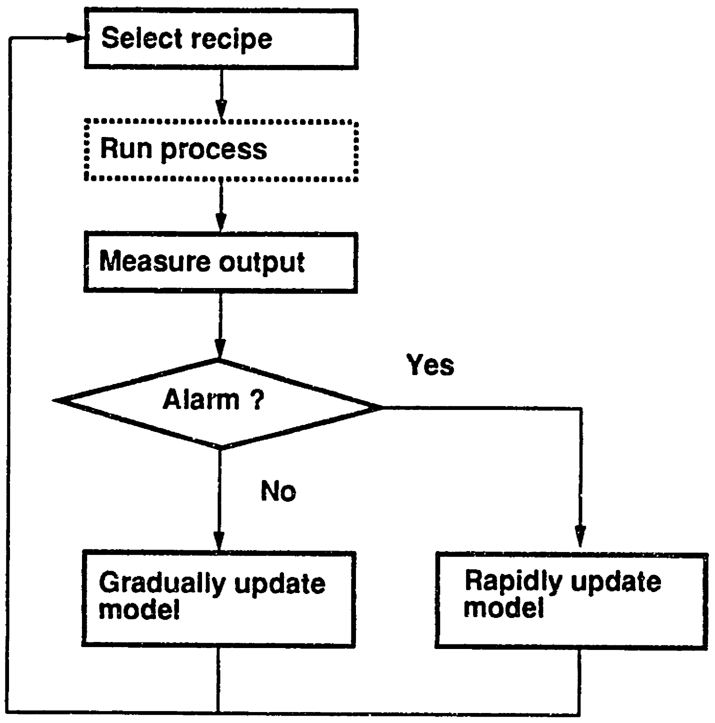


Figure 1-2: A flowchart showing the major components of the RbR controller algorithm.

- The process is drifting slowly but steadily, say on the order of 1 σ (the standard deviation of the process noise, to be defined precisely later) over a period of 100 runs.

- The process is subject to occasional, large shifts, say on the order of 3 σ.

These two situations are sufficiently different in character that the appropriate control strategies are different. For a slowly drifting process, the drift between successive runs may be one or two orders of magnitude smaller than the effect of noise for that run. Drift may caused by a variety of forces acting on the process, notably the aging of components, all of which act to direct the process away from target, but none of which is significantly greater than the others. Since the forces act at every run, and their effects tend to cancel each other at successive runs, the appropriate strategy for compensating for drift is to filter the output sequence, putting a relatively small weight on the most recent data, to get an estimate of the current level of the process. In other words, control should be gradual, and care should be taken to avoid over-control.

In contrast, for a process that is subject to occasional large shifts, that are an order of magnitude larger than the process noise, the appropriate controller would be to take no action unless the process was far from target, in which case the control action should be decisive. If a shift has occurred, the process will remain far from target until aggressive action is taken. Therefore, when significant evidence exists that a shift has occured, a more responsive control strategy is appropriate. This categorization of disturbances to a process is the basis for the two modes of the RbR controller: the gradual and rapid modes.

In the above discussion we have used the quantity σ without defining it precisely. In the traditional SPC paradigm, σ is the standard deviation of the output of a process in statistical control. But what if the process is never in perfect statistical control, i.e. what if there is always some drift present, as will usually be assumed in this thesis? The definition we will use is that σ is the standard deviation which the output would have if all disturbances that act to change the level of the process were compensated for exactly, before they had a chance to push the process off target. This, of course, is impossible (it requires looking into the future), so the actual standard deviation of the output will always be larger than σ.

The control strategy which has been described has broad applicability. However, the analyses performed in this thesis will be confined to the special case where all input variables are controllable and there is only one output variable, y_t. The performance criteria for run t will be the squared deviation of y_t from a target value T, i.e. $z_t = (y_t - T)^2$. Hopefully, the algorithms developed can serve as a foundation to build a strategy on for the general case of controlling multiple outputs of a process, whose dependency on the inputs may be non-linear, and where the performance criteria may include items such as the cost of adjusting the process and the cost of measuring the output variables.

For a batch manufacturing process, the values of the input variables might be changed *during* a run, in response to measurements made while the process is running, but it is not the purpose of the control algorithm described in this thesis to prescribe how such changes should be made. Rather, the algorithm suggests a recipe for each run, that may be then be modified during the run, as new information becomes available. This is why the algorithm will be referred to as the *Run by Run* (RbR) *Controller.* The activity of controlling the process during a run, in response to real-time measurements will be referred to as *real-time process control.*

The RbR controller will use the measured values of the input and output variables to continually refine a predictive model of the process. In postulating the predictive model, it will usually be assumed that the process function f_t is the sum of a linear function of the input variables and an error term ϵ_t:

$$y_t = \alpha_t + \beta' \mathbf{x}_t + \epsilon_t$$

The parameters α_t and β will be considered to be random variables. The intercept term α_t may change with time, but the vector β is assumed not to have time-dependent values. The assumptions made about the probability distribution of those parameters will then be used to obtain an appropriate prediction equation,

$$\hat{y}_t = a_{t-1} + \mathbf{b}' \mathbf{x}_t$$

which is used to select a recipe for the next run where the process output is likely to be close to target, i.e. a recipe $\mathbf{x}_t$ satisfying $a_{t-1} + \mathbf{b}'\mathbf{x}_t = T$. The values a_{t-1} and $\mathbf{b}$ in the prediction equation are estimates of the parameters α_t and β.

The vector of estimated process sensitivities $\mathbf{b}$ will be assumed to be available, for example from an off-line designed experiment. But the estimated intercept term a_{t-1} will be updated each time an output measurement y_t becomes available.

The output measurements y_t will also be monitored to decide whether the process is behaving in accordance with the current version of the predictive model of process behavior. As long as no radical departure from the predicted process behavior is signalled, the process model is updated gradually. But if the last few output measurements are in serious enough disagreement with their predicted values to signal an alarm, then this is taken as evidence that a significant disturbance to the process has occurred. In response to this evidence, the process model is updated rapidly, to allow the process to quickly adapt to the disturbance, and return the output to target.

1.2 Literature Review

Traditionally, there have been two main approaches to the type of process control which this thesis is concerned with: the *Statistical Process Control* (SPC) approach, and the *Automatic Process Control* (APC) approach. In recent years, the similarities and differences between these two approaches have been explored by various authors. In this section, some basic concepts underlying the application of SPC and APC will be described, and then papers by MacGregor ([Mac87]), Box and Kramer ([BK90]), a series of papers by Faltin and others ([WTFD90], [TFW90], [FHTW90], [Tuc90]), Baxley ([Bax91], [Bax90]), and Taguchi ([TEH89], [Tag87]) will be reviewed. All discuss issues pertinent to the interface between APC and SPC.

1.2.1 Statistical Process Control

Statistical Process Control is sometimes taken to include the use of any one of a variety of statistical methods to improve the performance of a manufacturing process. The SPC methodology considered here is more narrowly defined. The assumptions on which it is based are that the sequence of observations of a product characteristic, say $\{y_t\}$, is a random sample from a stable probability distribution. As long as the process behaves in this manner (i.e. as long as the process is *in control*), changing the inputs to the process will only increase the variability of the sequence $\{y_t\}$. However, if the values of the output sequence are not consistent with the postulated stable behavior, then one should look for *special causes* of change. If a cause of unstable behavior is found which increases the variability of the process. then it should be removed. If it decreases the variability, it should be made permanent. In statistical terms, each time a new value y_t becomes available, one tests the hypothesis that the process is operating in control.

The most common assumption is that the output sequence is a random sample from a normal distribution, centered on the target value T, with standard deviation σ. The values of the output sequence are plotted sequentially on a control chart, and the hypothesis of control is rejected whenever certain rules are satisfied, the most common one being that the current value y_t falls outside the interval $T \pm 3\sigma$. This type of control chart is known as a Shewhart chart ([Mon85]). Figure 1-3 shows an example of Shewhart control chart. The "three sigma" rule is designed to test the

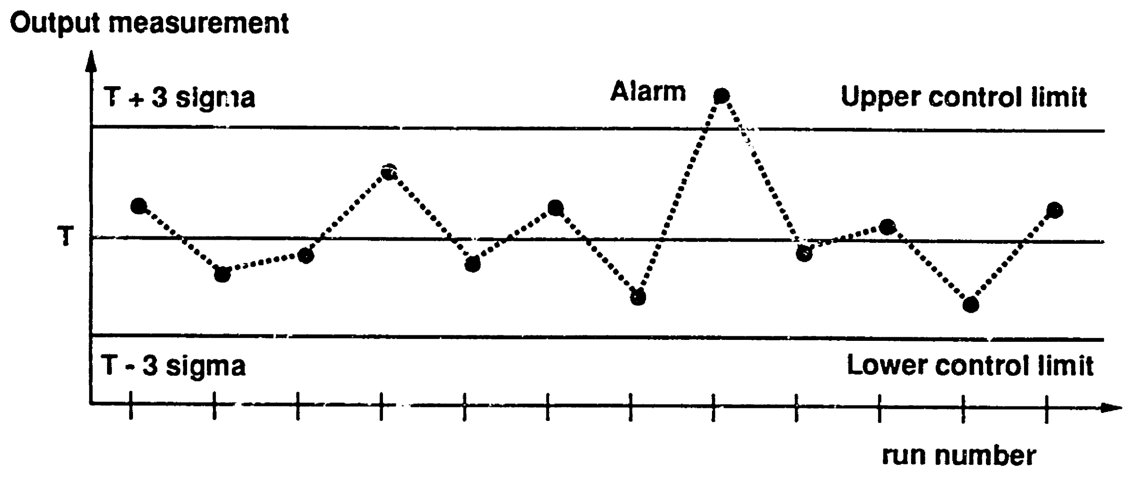


Figure 1-3: Control chart example

specific alternative hypothesis that the mean of the underlying distribution is the only parameter that has changed, i.e. the shape of the distribution has not changed and the independence between observations still holds. More powerful tests can be designed to test this limited alternative hypothesis, for example the CUSUM chart, which is a form of a sequential probability ratio test. But the advantage of using a Shewhart chart, which displays the data in its original form, is that an alert operator may be able to detect unexpected departures from the in-control state, for example a periodic pattern.

When it is possible with reasonable effort to find and eliminate assignable causes of disturbances, this is the preferred response. However, many commonly encountered circumstances are not amenable to the SPC paradigm. Examples include:

- The assignable causes are known, but it is either impossible or very expensive to remove them. For example, raw material variability may be very difficult to reduce. A different example is a maintenance operation, which can change the process behavior. If the change is predictable, feed-forward control could be used to compensate for it, but often this will not be the case.

- A process is undergoing slow drift. The drift might be due to known causes such as build up of deposition on the inside of a reactor, or it may be due to causes which are not precisely identified, such as the aging of components. SPC is not well suited to controlling a slowly drifting process for the following reason: under a no tweaking policy, the process must drift a certain distance before control action is taken, in response to an alarm. But if an inexpensive control action is available, then there is no reason to wait until the process has drifted "far enough". In addition, SPC does not specify what the control action should be.

In cases such as these, feedback control of the process has the potential to dramatically reduce the process variability. This is the motivation for developing the Run by Run controller: to efficiently regulate a process using feedback control, while using the diagnostic capability of a generalized version of SPC to detect sudden shift disturbances to the process. In addition, the RbR controller has the capability to quickly bring the process back on target after a shift disturbance.

Often, an automatic mechanism for changing the input variable is already in place, causing the marginal cost of adjustment to be low. Supposing that the standard response to an alarm is simply to adjust the input variable, one may wonder whether the hypothesis testing paradigm is appropriate, i.e. whether *significant* evidence of a process disturbance is necessary before a process adjustment can be justified.

To explore this issue, let us assume that a process is in control for extended periods of time, but occasionally, the mean of the process shifts. Let us also make the following critical assumption:

Assumption: When shifts occur, they are detected quickly by the control chart, and the resulting process adjustment successfully brings the process back on target.

If this assumption were true, and the shifts were infrequent, then the contribution of the shifts to the long term variation of the output would be small. Now suppose that the choice is between two control strategies: To adjust after every run on the one hand and adjusting only when the output falls outside the 3 σ control limits on the other hand, and suppose that our assumption holds for both strategies. The question is, which of the two strategies would result in smaller output variance? In a recent paper, Fellner [Fel91] argues that when an EWMA (Exponentially Weighted Moving Average) control rule decides on the size of adjustments, the asymptotic variance of the output is approximately the same whether small adjustments are made at every run of the process or adjustments are made only when significant evidence has been accumulated indicating that the process is off aim.

This result would seem to refute the argument that making small adjustments after each measurement inflates the variation of the output, compared to responding only when the output is outside the control limits. Indeed, it appears that if the size of the adjustment is determined appropriately, it does not matter how frequently adjustments are made.

But this conclusion depends on the assumption that the process is in statistical control for extended periods, and on the assumption we made about how the control strategy responds to shifts and this assumption is questionable. Indeed, if the shift is detected quickly, that implies that only a few measurements of the process after the shift are available. In turn, this implies that any estimate of the magnitude of the shift will be imprecise, and as a result the control action taken is unlikely to bring the process exactly on target.

If we decide that the assumption we made is unreasonable, then the question becomes: which control strategy minimizes the contribution of shifts to the output variation?

1.2.2 Automatic Process Control

In contrast to SPC, Automatic Process Control focuses on explicitly modeling the process dynamics and the disturbances to the process. Based on these models, a sequence of values for the input variables, say $\{\mathbf{x}_t\}$, are selected so as to satisfy specific design requirements. The control procedure is completely specified, hence the term Automatic Process Control.

The most important design requirements for an automatic control system are that the controlled process be stable and that it be sufficiently responsive. The stability requirement ensures that if the process is disturbed, it will eventually get back to target. The responsiveness requirement specifies how quickly the process should get back to target.

In this thesis, we will primarily be concerned with a particular form of APC, namely feed-back control. A feed-back controller tries to keep a process on target by acting on the error signal: the difference between the output and target. It selects the recipe $\mathbf{x}_t$ as some function of the error sequence $e_1, e_2, ..., e_{t-1}$, where $e_i = y_i - T$. A commonly used feed-back controller is the PID (Proportional - Integral - Derivative) controller. For the continuous time case, the PID controller (for a single input) takes the form of a linear combination of the instantaneous value $e(t)$ of the error, its derivative $\frac{d}{ds}e(s)\Big|_{s=t}$, and the integral of the error $\int_0^t e(s)ds$. Symbolically,

$$x(t) = k_p e(t) + k_d \frac{d}{ds}e(s)\Big|_{s=t} + k_i \int_0^t e(s)ds$$

The discrete time analogue of a PID controller is one where the derivative is replaced by the first difference $e_{t-1} - e_{t-2}$ and the integral is replaced by the sum $\sum_{i=1}^{t-1} e_i$:

$$x_t = k_p e_{t-1} + k_d(e_{t-1} - e_{t-2}) + k_i \sum_{i=1}^{t-1} e_i \tag{1.1}$$

Optimal Control

In certain idealized cases, one may be able to design a control system that is optimal, with respect to some criteria. An example of this is if the process dynamics and disturbances can be described using linear, constant coefficient difference equations. In particular, for the single-input-single-output case, the value of the output at time t is assumed to be a linear function of past values of the input variable sequence and a sequence of iid random variables, $\{v_t\}$.

To simplify notation in this subsection, instead of considering the relationship between the input x_t and output y_t, we will look at the relationship between the deviation of y_t from target, i.e. the error e_t, and the difference between the last two input settings x_t and x_{t-1}, which will be called $u_t = x_t - x_{t-1}$. It should also be mentioned that the time index t is used differently in this subsection than it is in the rest of the thesis: Here, x_t can be a function of y_t, whereas later it will be assumed that y_t is not measured until after x_t has been set, and the batch has been processed at that setting. In other words, in this subsection, a run starts with y_t being measured, and ends with x_t being set, but in the rest of the thesis, a run starts with x_t being set, and ends with the measurement of y_t. This discussion is based on [Ber87]. This material is also treated in [BJ76], for example.

The system can be economically represented using the backwards operator B, defined by $B^k u_t = u_{t-k}$. Define the following polynomials in B:

$$\begin{aligned}\psi(B) &= 1 + \psi_1 B + \cdots + \psi_m B^m \\ \omega(B) &= \omega_b B^b + \cdots + \omega_m B^m;\ 1 \le b \le m \\ \theta(B) &= 1 + \theta_1 B + \cdots + \theta_m B^m\end{aligned}$$

In terms of these polynomials, the system can be represented by

$$\psi(B)e_t = \omega(B)u_t + \theta(B)v_t \tag{1.2}$$

or, when written out in full:

$$\begin{aligned}e_t &= -\psi_1 e_{t-1} - \cdots - \psi_m e_{t-m} \\ &\quad +\omega_b u_{t-b} + \cdots + \omega_m u_{t-m} \\ &\quad +v_t + \theta_1 v_{t-1} + \cdots + \theta_m v_{t-m}\end{aligned}$$

From this, b is seen to be the delay with which a change in x affects y. The noise sequence $\{v_t\}$ is assumed to be a white noise sequence, i.e. the v_t are un-correlated random variables with $\mathrm{E}[v_t] = 0$ and $\mathrm{E}[v_t^2] = \sigma_v^2 < \infty$.

If the costs of adjustment and measurement are insignificant, then a reasonable control objective is to minimize the mean squared deviation of the output from target $\mathrm{E}\left[\sum_{t=1}^{N}(y_t - T)^2\right] = \mathrm{E}\left[\sum_{t=1}^{N} e_t^2\right]$ over some time horizon N. By the certainty equivalence principle [Ber87] an optimal control law with respect to this objective for the system described by 1.2 can be obtained by setting all random variables that have not yet been realized equal to their expected values, conditional on the currently available data. Let $D_t = (y_t, y_{t-1}, \ldots, x_{t-1}, x_{t-2}, \ldots)$ denote the data available at time t. Then the optimal controller is the solution to the set of equations $\{\mathrm{E}[e_t|D_t] = 0, t = 1, \ldots, N\}$ for u_t. But to take the expectation, one would need to know the distribution of $\{v_t\}$. However, even if this distribution is unknown, an estimator $\hat{e}_t$ can be derived [Ber87], satisfying

$$\hat{e}_t \to \mathrm{E}[e_t|D_t] \text{ as } t \to \infty$$

To this end, define polynomials $F(B) = 1 + f_1 B + \cdots + f_{b-1}B^{b-1}$, and $G(B) =$

$g_0 + g_1 B + \cdots + g_{m-1} B^{m-1}$, by

$$\theta(B) = \psi(B)F(B) + B^b G(B) \tag{1.3}$$

Define $\bar{\omega}(B)$ by $\omega(B) = B^b \bar{\omega}(B)$, so the system evolution equation 1.2 can be written as

$$\psi(B)e_{t+b} = \bar{\omega}(B)u_t + \theta(B)v_{t+b} \tag{1.4}$$

Next, multiply 1.4 with $F(B)$ and use 1.3 to get (after some manipulation)

$$\theta(B)\left[e_{t+b} - F(B)v_{t+b}\right] = \bar{\omega}(B)F(B)u_t + G(B)e_t$$

Since $F(B)$ has degree $b-1$, the expected value of $e_{t+b} - F(B)v_{t+b}$, conditional on D_t, is e_{t+b}. Therefore, a reasonable estimator is defined by

$$\theta(B)\hat{e}_{t+b} = \bar{\omega}(B)F(B)u_t + G(B)e_t \tag{1.5}$$

and the corresponding controller is obtained by setting $\hat{e}_{t+b} = 0$:

$$\begin{aligned} u_t &= \frac{-G(B)}{\bar{\omega}(B)F(B)} e_t \qquad (1.6)\\ &= -\eta_1 u_{t-1} - \cdots - \eta_{m-1} u_{t-(m-1)} \\ &\quad -g_0 e_t - g_1 e_{t-1} - \cdots - g_{m-1} e_{t-(m-1)} \end{aligned}$$

where $\eta_1 B + \cdots \eta_{m-1} B^{m-1} = \bar{\omega}(B)F(B)$. The resulting closed loop system is illustrated in figure 1-4. In terms of the input and output sequences, the controller can be represented as

$$\begin{aligned} x_t &= x^* - \eta_1(x_{t-1} - x_{t-2}) - \cdots - \eta_{m-1}(x_{t-(m-1)} - x_{t-m}) \\ &\quad -g_0(y_t - T) - g_1(y_{t-1} - T) - \cdots - g_{m-1}(y_{t-(m-1)} - T) \end{aligned}$$

Two comments about the controller defined by 1.6 are in order. First, it is implicitly assumed that the dynamics of the process and the disturbances, as defined by the polynomials ψ, ω, and θ, are known exactly. For a real manufacturing process, this is unlikely to be true. In chapter 6 of this thesis, the effect on stability of not knowing the process parameters exactly is analyzed for certain processes.

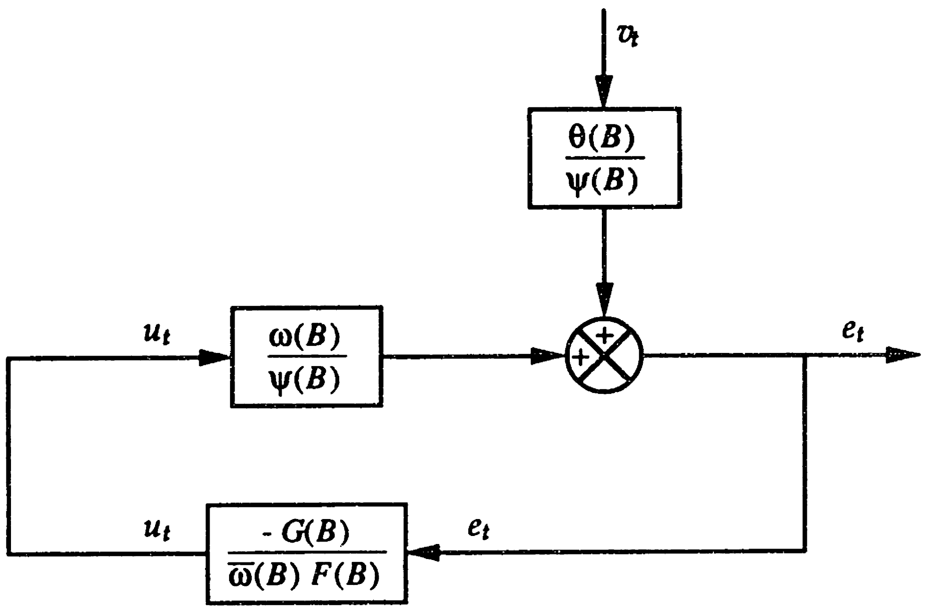

v_t
$\frac{\theta(B)}{\psi(B)}$
u_t
$\frac{\omega(B)}{\psi(B)}$
+
+
e_t
u_t
$\frac{-G(B)}{\overline{\omega}(B)\, F(B)}$
e_t

Figure 1-4: Block diagram of the closed loop system.

The second comment concerns the use of on-line SPC to complement the feedback controller 1.6. By substituting the controller in 1.2, which defines the behavior of the process, the closed loop process can be shown to be described by $e_t = F(B)v_t$. In the special case when the effect on the output of a change in the input is realized in the next interval (i.e. $b = 1$), the polynomial $F(B)$ equals 1. Then the closed loop process reduces to $e_t = v_t$. This means that as long as the process behaves as postulated, the control error sequence $\{e_t\}$ will be a random sample from some distribution. Therefore, on-line SPC can be applied to the control error sequence, to detect departures from the postulated process behavior.

Example

As an example of the application of the optimal controller 1.6, consider the deposition of silicon vapor (CVD) on silicon wafers. This process is part of the manufacture of Very Large Scale Integrated (VLSI) circuits. Figure 1-5 shows a reactor in which this process might be performed. The wafers are placed in a tray which is pushed into the reactor tube. A gas, such as silane (SiH_4) is pumped into the reactor chamber, that has been heated to around 1000° C. The gas reacts at the surface of a wafer and the result of the reaction is a deposited layer of polycrystalline silicon. It is important to

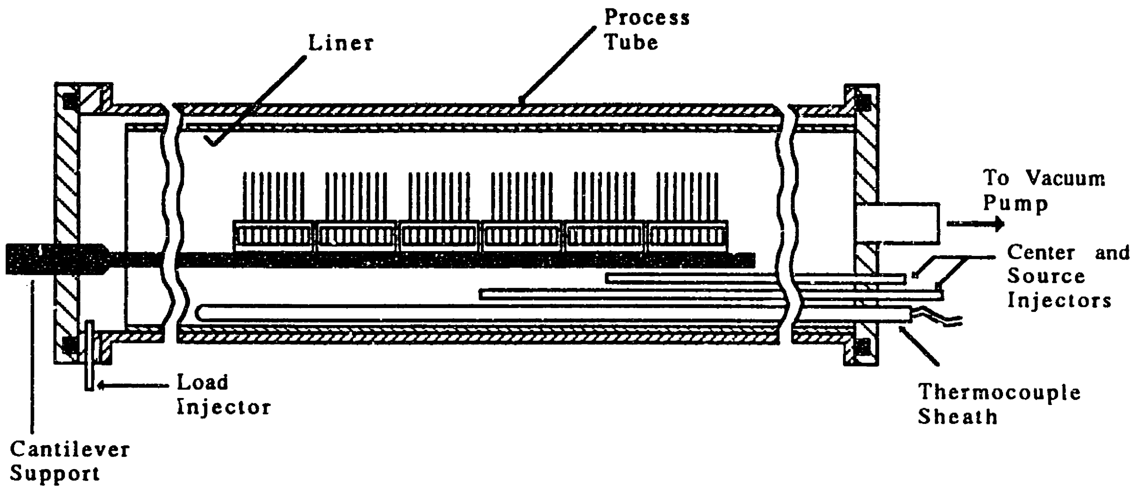


Figure 1-5: A diagram of a Low Pressure Chemical Vapor Deposition (LPCVD) reactor.

control the rate at which this layer is grown. Call the deviation of the growth rate from its target value e_t. An obvious way to control the growth rate is by varying the gas flow rate, which will be called x_t. One would expect that the larger the flow rate, the faster would the layer grow. Over a small range of x_t, it might be reasonable to model this relationship as a linear function:

$$y_t = \alpha_t + \beta x_t$$

Since the reaction chamber is allowed to cool down before the next batch of wafers is loaded in, there is no reason to believe that y_t should depend on values of x before time $t-1$.

But this process is subject to disturbances, which corrupt the simple relationship just given. The disturbances are of three major types:

1. Measurement error arising from the thickness measurement.

2. A gradual drift in the average level of y for a given x, caused, for example, by deposition on the walls of the reactor.

3. Sudden, large changes in the average level of y, given x. These step changes

could be caused by a maintenance operation, that might include cleaning the deposited layer from the reactor walls.

It might be reasonable to model the effect of the first two types of disturbances by the following process model:

$$\begin{aligned} y_t &= \alpha_t + \beta x_{t-1} + \epsilon_t \\ \alpha_t &= \alpha_{t-1} + \delta_t \end{aligned}$$

where ϵ_t represents noise, assumed to have mean zero and variance σ^2, and δ_t represents drift, assumed to have mean zero and variance $(r\sigma)^2$. By differencing, and using the identity $(1-B)y_t = (1-B)e_t$, the following difference equation is obtained:

$$(1-B)e_t = \beta u_{t-1} + \delta_t + \epsilon_t - \epsilon_{t-1}$$

Now assume that δ_t and ϵ_t are related via $v_t = \delta_t + \epsilon_t$ and $(1-\lambda)v_{t-1} = \epsilon_{t-1}$, for some $0 < \lambda < 1$. On taking the variances of both sides of these two equations, one obtains $\sigma_v^2 = (1+r^2)\sigma^2$ and $(1-\lambda)^2 = 1/(1+r^2)$. But with these substitutions, the process model is of the same form as the system equation 1.2, i.e.

$$(1-B)e_t = \beta u_{t-1} + (1-(1-\lambda)B)v_t$$

This special case of 1.2 is important not only because it represents a drifting process observed with noise, but MacGregor ([Mac76]) also shows that it is the limiting case of sampling the more general process with a longer and longer sampling interval. The estimator, $\hat{e}_{t+1}$ for this system is

$$\hat{e}_{t+1} = (1-\lambda)\hat{e}_t + \lambda e_t + \beta u_t$$

and the controller is obtained by setting the value of the estimator to zero:

$$u_t = -\frac{1}{\beta}\lambda e_t$$

The resulting closed loop system is simply $e_t = v_t$.

In terms of the actual outputs and inputs, the controller is

$$x_t = -\frac{1}{\beta}\lambda(y_t - T) + x_{t-1} \tag{1.7}$$

which can also be written as the integral controller $x_t = -\frac{\lambda}{\beta}\sum_{i=1}^{t} e_i$, and the closed loop system is $y_t = T + v_t$. This output sequence satisfies the assumptions necessary for the application of on-line SPC, as long as no disturbances of the third type described above occur. Therefore, a good way to detect such disturbances would be to plot the sequence $\{y_t\}$ and interpret it as a Shewhart control chart. □

Three comments will be made about this example:

1. The predictor $\hat{e}_t$ has the form of an Exponentially Weighted Moving Average (EWMA) of the previous values of the control error.

2. If control action is not taken at every run, then the predictor $\hat{e}_t$ should be updated at every run, and when control action is taken, its magnitude should be $-\hat{e}_t/\beta$. In this case, the closed loop system will *not* be $e_t = v_t$, so applying control charting techniques to the output directly would not be valid. But the one step ahead prediction error $\hat{e}_t - e_t$ can be shown to be equal to v_t in this case. Therefore, the sequence of prediction errors could be plotted on a Shewhart chart to monitor the state of the process.

3. This controller is only optimal over the class of linear controllers of the form $u_t = -\frac{1}{\beta}ce_t$, where c is some constant. In particular, if the disturbances to the process are not normally distributed, the globally optimal controller may not have this form. The meaning of the word "optimal" here should be clarified: One might argue that the optimal controller was the one that caused the output to be on target at every run. But this would require knowing the magnitude of random disturbances before they occur, which is impossible. Therefore, any reference to an optimal controller should be understood as a control law that specifies the input at time t as a function of the information available at that time.

1.2.3 Paper by MacGregor ([Mac87])

This paper describes typical applications of both SPC and APC. The paper is aimed at readers that are knowledgeable about process dynamics and automatic control, but have had little exposure to statistical methodology. MacGregor attempts to put the relative merits of the two approaches in perspective using the theory of optimal control of linear processes, as presented in the preceding subsection. He considers the case when there are costs associated with adjusting the input, and it is desired to minimize the sum of these costs and the cost of the output variance. It turns out that here, the optimal control strategy is to take control action only when the predicted deviation (using the predictor 1.5) of the output from target becomes larger than some threshold (i.e. goes outside a "deadband"), and then adjust the process so that the predicted deviation for the next run will be zero.

In the case where the output depends only on the last input setting (i.e. $\omega_{b+1} = \cdots = \omega_m = 0$), this control law reduces to monitoring a filtered version of the output sequence, and taking control action whenever the filtered sequence goes outside a set of limits. While this scheme appears similar to on-line SPC control, the interpretation of an "alarm" is different. An SPC alarm is interpreted as evidence that an unanticipated external disturbance to the process which has changed the behavior of the process has occurred. In contrast, when the predicted output deviation from target reaches its deadband limit, this is not taken as evidence of any change in process behavior. It simply means that the process has drifted far enough from target that it is now cost efficient to take a control action.

1.2.4 Paper by Box and Kramer ([BK90])

In contrasting SPC with APC, Box and Kramer argue that

- The users of SPC and APC usually have different backgrounds.
- The two methodologies developed under different circumstances, with SPC originating in the discrete parts industries abut APC in the continuous process industries.
- As a result, the control objectives and the (sometimes tacit) assumptions on which the two methodologies are based are different.

However, new industries, such as the semiconductor industry, have attributes of both the parts and process industries. Therefore, the choice of methodology for effectively controlling a process in these industries is not always obvious.

For any process, the three major issues that determine the appropriateness of a control strategy are the disturbances to the process, the dynamic (or inertial) characteristics of the process, and the costs of operating the process. Box and Kramer present the following special case of the linear process model that has been discussed already, which they argue can capture the behavior of a variety of processes:

$$y_t = c + \delta y_{t-1} + \beta(1-\delta)x_{t-1} + v_t + \lambda \sum_{i=1}^{t-1} v_i$$

where $0 \leq \lambda \leq 1$ and $0 < \delta < 1$. In fact, the model presented in the LPCVD example is a special case of this model, with $\delta = 0$, corresponding to no process dynamics. Here λ is a measure of how quickly the process is drifting, δ is a measure of the process inertia, and β is the process gain. For this model, the optimal control law turns out to be a PI (Proportional - Integral) controller, with the coefficients (cf. equation 1.1)

$$k_p = \frac{\lambda\delta}{\beta(1-\delta)} \text{ and } k_i = \frac{\lambda}{\beta}$$

If the process were in statistical control, the non-stationarity parameter γ would equal zero, and both of the above coefficients would be zero, implying that taking no control action is optimal. The authors then consider the case when there is a significant cost associated with making adjustments (as MacGregor did) and also with measuring the process output. As in the case considered by MacGregor, the optimal controller turns out to be a deadband controller.

Box and Kramer also address some criticisms that have been directed at SPC and APC, respectively. For example, it sometimes claimed that APC over-compensates, as compared to SPC. This is of course true if the process is really stationary, in which case no control is optimal. But if the optimal controller is used, based on reliable data about the characteristics of the process, then by definition it will not over-compensate.

1.2.5 Algorithmic Statistical Process Control

The series of papers ([WTFD90], [TFW90], [FHTW90], [Tuc90]) describe an approach to process control which the authors c.." *Algorithmic Statistical Process Control* (ASPC). ASPC represents an integrated approach to quality improvement, which is based on regulating the process using APC and at the same time using the diagnostic capability of SPC to find and eliminate root causes of variability. In this context, SPC would serve the strategic purpose of improving the process, while APC would be in the tactical role of effective adjustment. The authors use as an analogy the process of driving a car: The driver needs to constantly adjust the direction and speed of the car (the role of APC) and at the same time be alert to signals of an engine malfunction or a flat tire (the province of SPC). As summarized in [WTFD90], "You can't drive a car with SPC and you can't 'fix' it with automatic control."

The following general guidelines are given for implementing ASPC:

1. Identify and estimate the characteristics of the process and disturbances.
2. Design an APC control law, based on a model of the process and pertinent costs.
3. Design an SPC monitoring scheme to monitor the closed loop system.
4. When an alarm occurs, search for assignable causes, or repeat the process identification and estimation stage, if necessary.

1.2.6 Papers by Baxley ([Bax91], [Bax90])

In these two papers, Baxley presents an EWMA control algorithm that has both a deadband and control limits. When the EWMA predictor goes outside the deadband, control action is taken to bring it back to zero, and when the predictor goes outside the control limits, a search for assignable causes is started. The algorithm is as follows:

1. After each run, measure the devation from target e_t. The EWMA statistic $\hat{e}_{t-1}$ is a predictor of this deviation, so the forecast error is $e_t - \hat{e}_{t-1}$. If the forecast error is inside the control limits, i.e. if $|e_t - \hat{e}_{t-1}| < K\sigma_e$, then update the EWMA predictor using

$$\hat{e}_t = \lambda e_t + (1 - \lambda)\hat{e}_{t-1}$$

Otherwise, if the forecast error is outside the control limits, the EWMA predictor is not updated, but a search for assignable causes of variation is started.

2. If the EWMA predictor is outside the deadband limits, i.e. $|\hat{e}_t| > L\sigma_z$, then adjust the process to target by setting $x_t = x_{t-1} - \hat{e}_t/\beta$, (where β is the process gain), and reset the predictor $\hat{e}_t$ to zero. Otherwise, do not take any control action.

Note that when there is no deadband (L = 0), then this controller is identical to the one given by 1.7 in the LPCVD example. Except for the control limits, this algorithm would be the same as the one advocated by MacGregor, and Box and Kramer for the case when the cost of adjustment is taken into account, if the deadband limits were chosen appropriately.

But Baxley does not dwell on whether this algorithm is optimal or not. In contrast to the papers menitoned so far, he does not assume that the parameters of the process (e.g. the process gain β and the EWMA parameter λ) are known exactly. Using simulation, he explores the effect of using suboptimal values for these parameters and concludes that the controller is robust to minor variations in these parameters.

1.2.7 Taguchi's "Beta coefficient" Method

Genichi Taguchi describes his "beta coefficient method of on-line process control in [Tag87, chapter 19] and [TEH89, appendix B] (Since we have already used the symbol β for the process sensititivy, we will use the symbol c for the quantity Taguchi calls beta). He starts his development by assuming that the process behavior is such that the process level drifts randomly between two successive runs, and that the measured output is the process level, corrupted by random noise. These assumptions are identical to the ones made in the LPCVD example. Taguchi presents his control strategy as an alternative to a "deadbeat" controller, which in the notation of the LPCVD example would be defined by $u_t = -\frac{1}{\beta}e_t$. In other words, the deadbeat controller compensates for the total deviation e_t, rather than scaling it down to λe_t. The closed loop system under deadbeat control is $e_t = (1 - (1-\lambda)B)v_t$, and the variance of the deviation e_t can be shown to be $100(1-\lambda)^2$ percent larger than it would be with the optimal controller $u_t = -\frac{1}{\beta}\lambda e_t$.

Taguchi suggests using the following control law:

$$u_t = -\frac{1}{\beta}\max\left(0, \frac{e_t^2 - \sigma_v^2}{e_t^2}\right) e_t$$

In deriving this control law, Taguchi's objective is not to minimize the variance of the deviation e_t of the output y_t from target, but to minimize the deviation of the underlying level from the target T. By the "underlying level" is meant the output uncorrupted by the noise term ϵ_t, i.e. $y_t - \epsilon_t$. Unfortunately, the underlying level is not observed directly, so Taguchi is forced to approximate. In particular, he approximates the squared devation of the underlying level $(y_t - \epsilon_t - T)^2$ by its expected value, which is $(y_t - T)^2 - \sigma_v^2$.

It might be argued that Taguchi's objective of keeping the underlying level, rather than the observed output, on target is more appropriate, since the quality of the product depends on its true characteristics, rather than the measured characteristics. But there are some questions and concerns that can be raised about the appropriateness of Taguchi's method, for example:

1. What is the effect of approximating $(y_t - \epsilon_t - T)^2$ by $(y_t - T)^2 - \sigma_v^2$?

2. The term ϵ_t might represent not only measurement error, but also errors in modelling. For instance the dependency of the output on the input might be quadratic rather than linear, in which case it is not clear which of the "underlying level" and the measured output is a better measure of the true characteristics of the product.

A simulation study performed by Mittmann [Mit91] suggests that for slowly drifting processes, the EWMA algorithm of the gradual mode of the RbR controller consistently outperforms the beta coefficient method.

1.3 Remainder of the Thesis

The Run by Run Controller was developed as part of a modular system for process control in VLSI (Very Large Scale Integrated) circuit fabrication. Some of the relevant characteristics of VLSI manufacturing processes, and the process control system will be described in chapter two.

The third chapter presents an overview of the algorithm. The details of the gradual and rapid modes are described in chapters 4 and 5, respectively. In chapter 6, the stability and robustness of the algorithm will be analyzed, and in chapter 7 a simulation experiment, whose purpose was to evaluate the performance of the algorithm, is reported on. In chapter 8 conclusions are made and future extensions are suggested.

Chapter 2

Setting

The RbR controller was designed as part of an overall process control system. The proposed process control system is designed with VLSI fabrication in mind, but the methods developed are general enough that they may be applicable to other kinds of batch processes.

VLSI circuits are usually made on silicon wafers. The silicon wafers will go through several processing steps before becoming a finished product. In this system, each process in the sequence is to be controlled separately. To accomplish this, measurement data from the current process and the process immediately preceding it are used.

The system consists of three main modules. The Flexible Recipe Generator determines the best initial operating conditions, given the product specifications and the available knowledge about the production process. The Run by Run Controller modifies the operating conditions for each new run, based on measurements of the current state of the process and more refined knowledge about its operation. The Real Time Controller makes adjustments during a production run.

A block diagram of this system is shown in figure 2-1. The system is described in detail in [SGHH91].

This chapter has two sections. The first describes the role of the RbR controller in the overall process control system. The second gives a very brief overview of the manufacture of VLSI circuits, with attention focussed on those processing steps where applying the RbR controller might yield the greatest benefits, and on justifying some of the assumptions made in modeling the system.

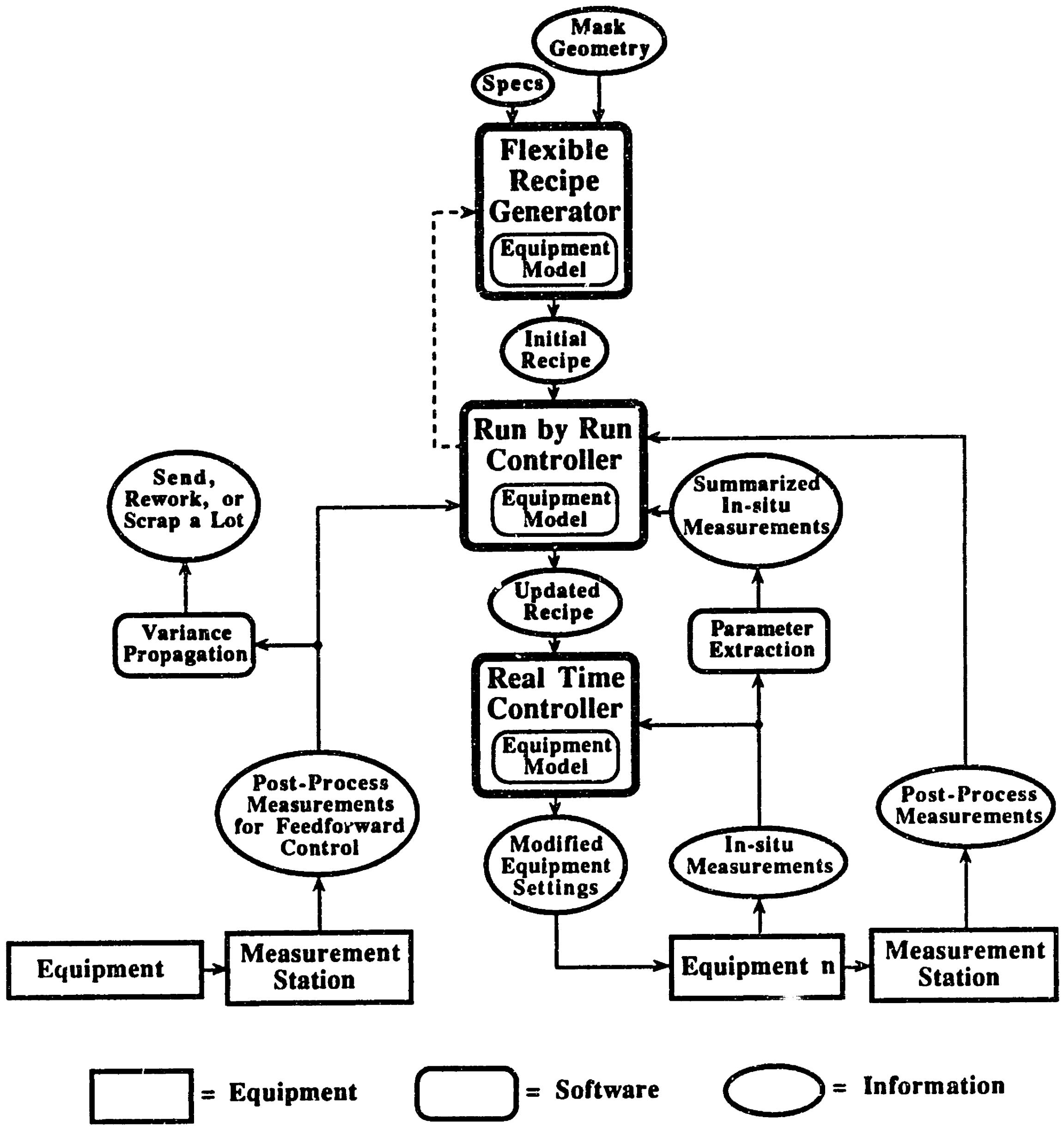


Figure 2-1: Block diagram of a process control system for VLSI fabrication

2.1 Role of the RbR Controller

The purpose of the RbR controller is to update the current recipe, based on current measurements of the process and of the preceding process. This is done using local on-line optimization, feed-back control, and feed-forward control.

Local on-line optimization refers to systematic exploration of operating conditions in the hope of achieving better performance. This exploration is constrained to a small region around the current recipe to ensure that the percentage of scrap is not increased.

Feed-back control is concerned with identifying disturbances to a process and adjusting the recipe such that the performance of the process is as good as before. Sometimes, the cause of the disturbance is known, a typical example being a maintenance operation. In other cases, a disturbance occurs for unknown reasons, but its existence can be inferred from the effect on the output y_t. Two useful categories of disturbances are shifts and drifts, referring to an abrupt change and a gradual change over a period of time, respectively. The RbR Controller algorithm attempts to compensate for drifts and identify shifts in a process and respond appropriately.

Feed-forward control is the activity of adjusting the recipe for the process based on measurements made on the output from the preceding process, to compensate for maladjustment in that process. Note that in order for feed-forward control to be useful, one must have a good understanding of the physics of the process (i.e. be able to predict its performance over a wide range of operating conditions).

The overall architecture of the RbR controller is illustrated in figure 2-2. The RbR controller responds to in situ and post-process measurements from a process with two modes of operation: optimization and control. Optimization may be repeated periodically, if it is thought that additional possibilities for improvement exist, or if the process behavior is thought to have changed drastically. Once the process has been optimized, the control mode is engaged in order to maintain the process at the optimum level. The control mode is the subject of this thesis.

The distinction between the two modes is that in the optimization mode, it is expected that the process can be significantly improved, while in the control mode, the concern is to maintain the performance of the process in the face of disturbances. Thus, in the optimization mode, improvement can be sought by aggressively exploring

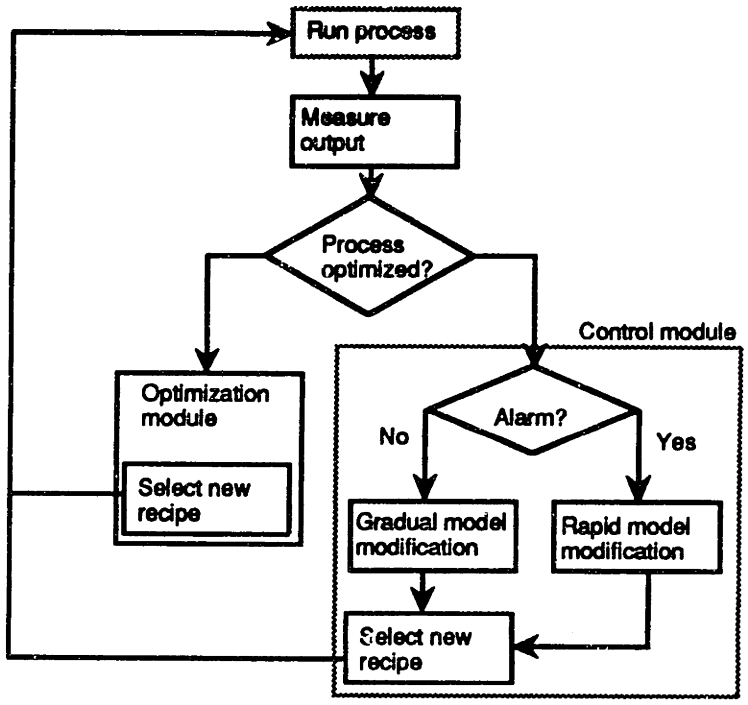


Figure 2-2: A flowchart showing the overall architecture of the Run by Run controller.

the process space even though doing so will temporarily increase the variability of the process output. In the control mode, on the other hand, the emphasis is on controlling to target. The optimization aspect of the RbR controller is described in [GSHH90] and [SGHH90].

The feed-back control module of the RbR controller (as opposed to the optimization module) interacts with the other modules in the process control system in the following ways (refer to figure 2-1):

- An initial recipe along with a predictive model of the process is obtained either from the flexible recipe generator, or from the optimization module of the RbR controller.

- A suggested recipe for each run is given to the real time controller.

- Summarized in-situ measurements are obtained after each run from the real time controller.

- Post process measurements are obtained after each run from a measurement station.

2.2 VLSI Fabrication

A Very Large Scale Integrated (VLSI) circuit, or a *chip*, consists of a network of electrical devices, such as transistors, which is constructed on a silicon substrate. The common definition of VLSI is a circuit with more than 100,000 devices per chip ([WT86]).

The manufacture of chips starts with the growing of a single crystal cylinder-shaped ingot of silicon. The ingot is sliced into thin wafers (675 μm is a typical thickness for a wafer which is 150 mm in diameter), on the surface of which the circuits are constructed. We will be concerned with the processing steps which occur after the silicon wafers have been polished and prepared, and before the individual chips are separated and packaged.

The circuit construction consists of growing several thin films (on the order of 1000 Å) of material on the surface of a wafer. The film material can be, for example, silicon oxide (SiO_2, an insulator), polycrystalline silicon (a semiconductor), or aluminum, for making contacts and wiring.

The patterning of each film follows a cycle, which usually consists of the following steps: First, the film is grown or deposited on the silicon substrate, by means of chemical vapor deposition (CVD) or thermal growth. Second, photolithography is used to create protective layers of photoresist over those areas of the film that should remain. Finally, the film material which is not protected by photoresist is etched away, leaving the desired film pattern.

In order to provide a context for the analysis performed in this thesis, one chip production step will now be described. For further detail, see [WT86].

Epitaxial Growth Process

The epitaxial growth process is a means of depositing a thin layer of single crystal silicon on the surface of a silicon wafer. The wafers are typically located on a hexagonally shaped susceptor inside a bell jar shaped reactor chamber (see figure 2-3). Gas (a mixture of trichlorosilane ($SiHCl_3$) and hydrogen) flows in through two nozzles at the top of the reactor and reacts at the surface of the wafers to form the deposited layer of silicon. It is critical for the proper functioning of the electronic devices on the wafer that the epitaxial layer be of uniform thickness. Suppose that for each batch of wafers loaded into the reactor, the layer thickness is measured on a monitor wafer at the five locations indicated in figure 2-3. A possible output measure of thickness uniformity is the difference between thickness measurements on the left and right sides of a wafer, which ideally should be zero. Another possible uniformity measure would be the sample variance of the thickness measurements. Factors which are known to have a major effect on the thickness uniformity are the balance of flow between the two nozzles, assuming a constant total flow rate and the horizontal angle at which the nozzles point into the reactor chamber.

2.2.1 Modeling Assumptions

Each step in the chip manufacturing process has an associated set of inputs and outputs. The purpose of process control is to manipulate the process inputs so as to cause the process outputs to have desirable values. In order to analyze the behavior of a control strategy, a model of the causal relation between inputs and outputs is needed. A model used in this thesis is based on the following assumptions:

- The process has no dynamics, in the sense that the outputs depend only on the input settings for the current run, not on settings at previous runs.
- The output for a given run is a linear (first order) function of the inputs for that run.
- The process sensitivities can be considered constant for extended periods of time.
- Disturbances to the process can be classified as either a drift or a shift in the process level.
- The variance of the inherent process noise (caused, for example, by measurement error) remains constant for extended periods.

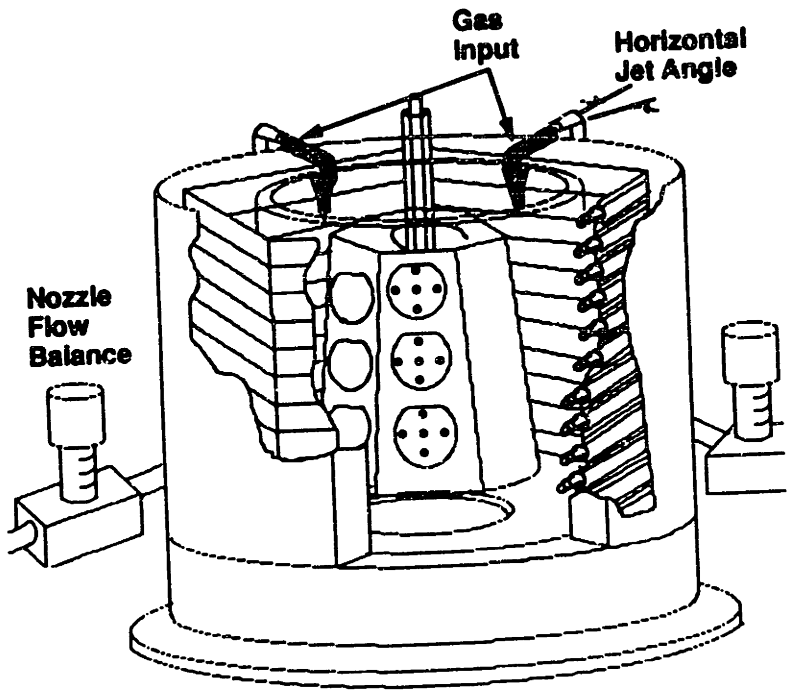


Figure 2-3: A schematic representation of a radially heated epitaxial reactor. Two control variables which can be used to control the thickness uniformity of the epitaxial layer are the balance of gas flow between the two nozzles shown at the top of the reactor and the angle between each of the nozzles and a line to the center of the barrel. Six silicon wafers are shown on the hexagonal susceptor. On three of them, five thickness measurement sites are indicated.

This model will be formalized in the next chapter. In the remainder of this section, the validity of these assumptions will be discussed.

Lack of Dynamics

In the CVD process described in the exmaple in section 1.2.2, a batch of wafers is loaded into a reactor, which is then heated up to the desired temperature. Gas is then pumped in, which reacts at the surface of a wafer, to form the deposited film. After the reaction is completed, the reactor is cooled down, and the wafers are removed before the next batch is loaded in. Since the reactor chamber is allowed to return to atmospheric conditions between batches, there is no reason to expect any carry-over effects. In other words, process outputs such as film thickness and thickness uniformity are a function of inputs such as temperature, pressure, and flow rates for the current run only, and is independent of previous runs.

Linearity

In reality, no manufacturing process is perfectly linear. But since a linear process is much easier to analyze and interpret than a nonlinear one, one hopes that within the region of operation, the process can be approximated by a linear function. In the next chapter, the stability of a process controlled by the RbR controller is analyzed, under the assumption that the process is quadratic rather than linear. It is shown that the condition which guarantees stability is similar to the linear stability condition.

The validity of assuming that a process is linear can also depend on which output variables are used. For example, in the epitaxial growth process, the left to right thickness difference can reasonably be expected to depend linearly on the balance of flow between the two nozzles, at least within some narrow operating range. For the sample variance, on the other hand, this claim is harder to make. In fact, the relationship between sample variance and balance of flow may not even be monotonic within the region of operation.

In [SPG91], a computer simulation based on physical models of a Low Pressure CVD (LPCVD) process is described. Figure 2-4 shows how the average layer thickness depends on the total gas flow rate for the simulated LPCVD process. The figure shows that when the flow rate ranges from 130 to 150 sccm, there is hardly any curvature in the relationship between thickness and flow rate.

Constancy of Process Sensitivities

This assumption is a critical one, and is not always warranted. A disturbance, such as a maintenance operation, which changes the behavior of the process, can change not only the level but also the sensitivities of the process. As shown in the next chapter, the process stability depends on the reliability of the estimated process sensitivities. Therefore, an important enhancement of the algorithm would be to allow for re-estimation of the process sensitivities. During normal operation (i.e. in the absence of shifts) the input settings are not likely to change very much, making estimation of the process sensitivities difficult. But when shifts occur, a larger change is made in the input settings, to compensate for the shift. Therefore, it might be possible to

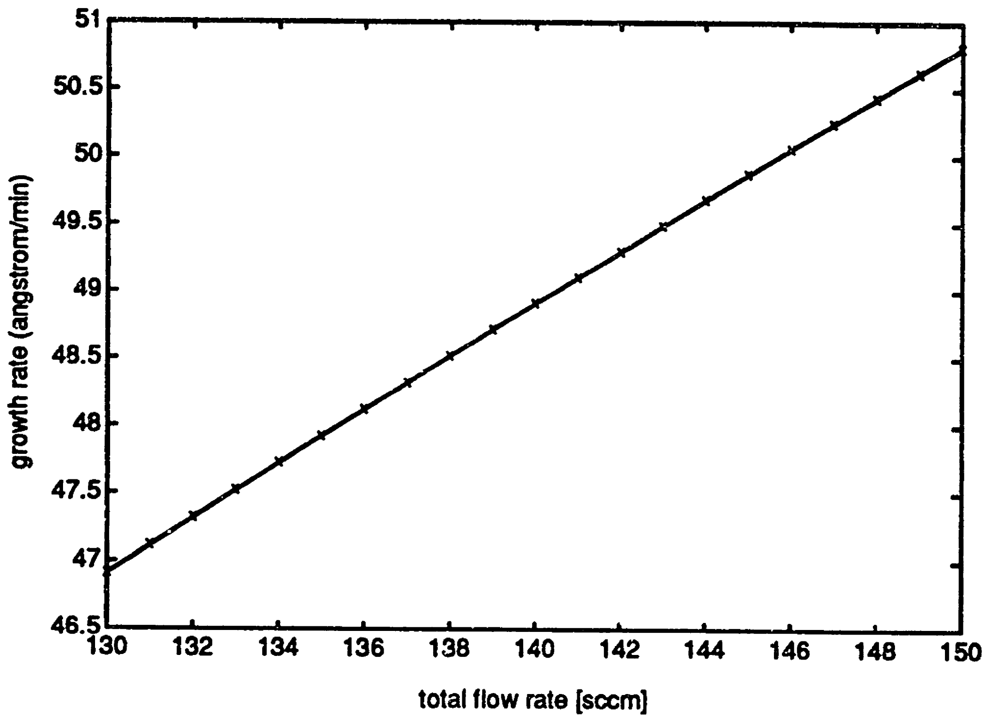


Figure 2-4: Data from an LPCVD simulator. The simulator was run with the total gas flow rate taking values betwen 130 and 150 sccm. The resulting average growth rates in Å/min. are shown. Observe that the relationship is very close to being linear.

get reliable information about changes in the process sensitivities from the runs just before and after compensation for a shift.

In [SHIL91], an experiment performed on the epitaxial growth process described previously is reported on. The results show that the process sensitivities did change significantly after a maintenance operation, and an algorithm which updates the sensitivities is shown to enhance convergence of the process to target after the maintenance operation.

Nature of Disturbances

Disturbances to a manufacturing process could have any number of different forms. However, many commonly encountered disturbances may be classified as contributing to either drift or shift in the process level. By drift we mean a slow but steady change, on the order of 1 σ over a period of 100 runs. By shift we mean occasional, large (on the order of 3 σ) changes in level.

Drift may be caused by slow changes in environmental conditions, such as temperature and humidity, deposition inside a reactor, the aging of components, drift in the mass flow controllers which maintain setpoints for gas flow, and a number of other causes. Shifts often have well known causes, such as a maintenance operation or the beginning of a new shift, but sometimes the time at which the shift will occur will be unpredictable. For example it may be known that for an etching operation, the etching through a certain layer inside the vessel in which the etching takes place will cause a shift, but it may be difficult to predict precisely when this will happen.

The choice of those two categories of disturbances is motivated also by the fact that they are sufficiently different in character that different control strategies are appropriate for each.

Chapter 3

Overview of the Algorithm

This chapter will give an overview of the Run by Run Controller algorithm. The generalized version of SPC used to detect major disturbances will be described. Also, the process model used to evaluate the performance of the algorithm analytically will be presented.

The goal of the algorithm is to maintain the performance of a manufacturing process in the face of unpredictable disturbances. This is done by continuously updating a model of the process, using input settings and process measurements. A high level summary of the algorithm is shown in figure 3-1. Before each run of the process, the algorithm selects a recipe $\mathbf{x}_t$ of input settings to be used. This recipe is chosen so that the predicted output $\hat{y}_t$ equals the target T, using the latest available model of the process, which is $\hat{y}_t = a_{t-1} + \mathbf{b}'\mathbf{x}$. The process operator may choose not to take the advice of the algorithm, in which case the predicted output may be different from T. Then, the process is run at the recipe chosen by the operator, and an output measurement y_t is obtained. The prediction error $\hat{y}_t - y_t$ is computed and plotted on a Shewhart control chart. If the control chart generates an alarm, this is interpreted as evidence that a step disturbance to the process has occurred. As a result, the algorithm enters the rapid model modification mode (*rapid mode*), wherein the predictive model of the process is updated quickly to allow rapid compensation for the step disturbance. On the other hand, if the measurements from the current run did not cause an alarm, and an alarm has not occurred in the immediate past, then the algorithm will operate in the gradual model modification mode (*gradual mode*). In gradual mode, the intercept term of the predictive model of the process is updated

```
Initialize process model, $\hat{y} = a_0 + \mathbf{b}'\mathbf{x}$.
$t \leftarrow 1$.
while control is needed do
    Select next recipe, $\mathbf{x}_t$.
    Predict response, $\hat{y}_t = a_{t-1} + \mathbf{b}'\mathbf{x}_t$.
    Operate process, measure actual response, $y_t$.
    Compute prediction error, $e_t = \hat{y}_t - y_t$.
    Decide whether a shift has occurred (based on $e_i$, $i = 1, 2, \ldots, t$).
    if a shift has occurred, then
        Estimate magnitude and location of shift disturbance.
        Adjust the process model.
    end if
    Update the process model.
    Let $t \leftarrow t + 1$.
end while
```

Figure 3-1: The Run by Run Controller algorithm

as an Exponentially Weighted Moving Average (EWMA) of the evidence supplied by previous runs of the process.

This algorithm can be interpreted as a generalized version of Statistical Process Control (SPC), as will now be described.

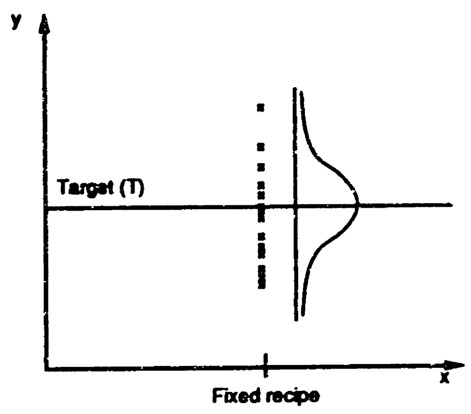
3.1 Generalized SPC

A basic assumption often made when SPC is applied is that the equipment settings must remain stable, that is they cannot be tweaked, because otherwise the control charts cannot be properly interpreted. However, equipment settings must often be tweaked to optimize the operation of equipment, and to respond to process shifts and drifts. In this section, a generalized version of SPC, which allows the process to be tuned and monitored at the same time will be described. Since the RbR controller changes the process recipe between runs, the generalized version of SPC is critical to its operation.

The traditional tool of SPC is the control chart, on which the process output y_t is plotted over time ([Mon85]). The statistical techniques used to diagnose control charts are based on the assumption that the values which are plotted are independent samples from a common probability distribution. Because of this assumption, it is generally thought that the inputs x_t, must be held constant in order for valid conclusions to be drawn from an analysis of a control chart, since otherwise the probability distribution from which y_t will change over time. The result is that no tweaking is permitted during the use of a control chart. This constitutes a serious restriction on the operation of a process.

Underlying SPC is a binary view of the condition of a process, i.e. either it is running satisfactorily or not. In reality, there is an opportunity for improving the process by optimizing it and by re-tuning it more frequently than does SPC to compensate for small shifts and drifts. The RbR controller can accomplish such optimization and re-tuning because it continually updates a model for the process as production continues.

Since the RbR controller must tweak the process as part of its operation, clearly the traditional approach to SPC cannot be used. To retain the benefits of SPC (the ability to diagnose a process) while tweaking a process, generalized SPC is being developed. In generalized SPC, the measured output value is compared with a predicted output value and the difference is plotted on an SPC chart which is normally used to plot the measured value itself. Generalized SPC is contrasted with the traditional version in figures 3-2 and 3-3. In traditional SPC, the measurements (or the average of a sample of measurements) obtained by running the process at a fixed recipe are plotted on a control chart, as shown in figure 3-2. When generalized SPC is applied, the recipe may be changed, in order to search for better performance, or to compensate for disturbances. If the process was originally in statistical control, then changing the recipe will in fact increase the scatter of the output measurements as shown in figure 3-3. Therefore, if the measurements were plotted directly on a control



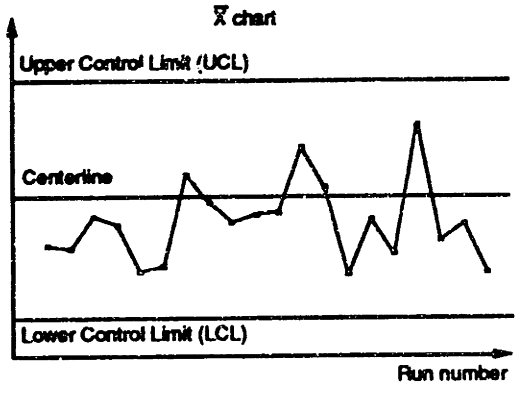


Figure 3-2: In the traditional application of SPC, the setpoint is fixed (the process is not tweaked). The output measurements (or the average of samples of measurements) are plotted on an $\bar{X}$ control chart. Assuming that the process is in statistical control, most of the measurements will fall between the upper and lower control limits, which are usually set at $T \pm 3\sigma$, where σ is the standard deviation of uncorrelated noise in the process.

chart (the black squares in figure 3-3), a large number of alarms would occur. But these alarms are caused not by a lack of statistical control, but because the recipe is not constant. To avoid this problem, instead of plotting the measurements, we plot the one step ahead prediction error for the process output. That is, we plot the difference between the measured value and the prediction of the process output based a model developed from the preceding data. The model used is the statistical model created and updated by the RbR controller. An early paper suggesting this strategy is ([Man69]).

Shewhart control charts are efficient at detecting large changes in the process mean, say on the order of 3 σ, but they are less efficient for detecting small changes. For this reason, different control charts, such as the CUSUM and EWMA control charts ([Mon85]) have been developed, to detect minor changes more quickly. However, the gradual mode of the RbR controller is designed to compensate for slow changes in the process mean. Therefore, detection of such changes is not particularly

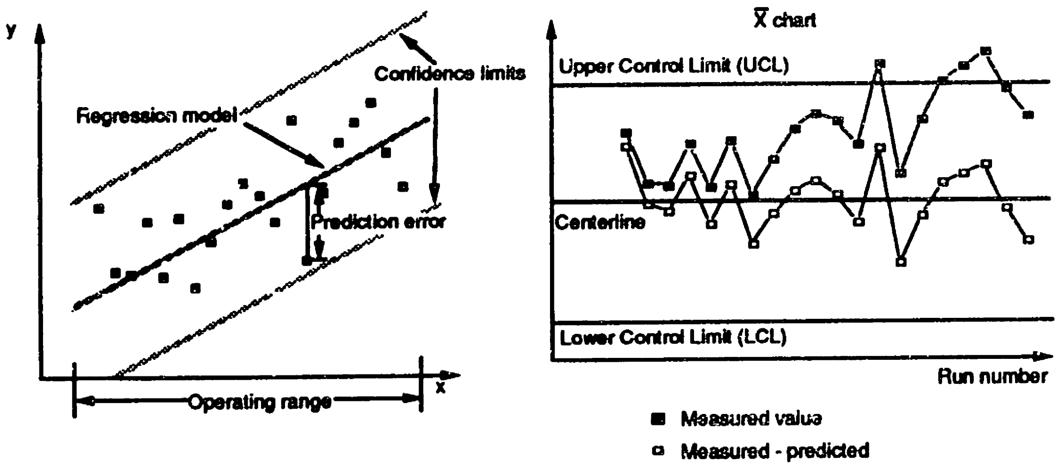


Figure 3-3: If a process is in statistical control, but a control variable which affects the output is being tweaked, then the output measurements will not be a random sample. This is illustrated by plotting the measured values (the black squares) on a control chart. Although the process is in control, several of the measurements are outside the control limits, because the value of the control variable is being increased with time. In generalized SPC, instead of plotting the measured values, the one step ahead prediction errors (difference between measured output value and the value predicted by a process model) are plotted. These values are shown as white squares on the control chart, and they are all inside the control limits.

important when the RbR controller is applied.

3.2 Process Model

In analyzing the behavior of the RbR controller algorithm, attention will be restricted to a model where the output at time t is the sum of function which depends only on the current recipe $\mathbf{x}_t$ and a finite dimensional parameter vector Θ_t; and an independent error term ϵ_t:

$$y_t = f_t(\mathbf{x}_t, \Theta_t) + \epsilon_t$$

In particular, we will focus most of the time on the linear model

$$y_t = \alpha_t + \beta' \mathbf{x}_t + \epsilon_t$$

where the intercept term α_t is a time dependent random variable, but the vector of first order coefficients consists of random variables whose distribution does not depend on time. The sequence $\{\epsilon_t\}$ is assumed to be a white noise sequence, i.e. the ϵ_t's are independent and identically distributed (iid) with $\mathrm{E}[\epsilon_t] = 0$ and $\mathrm{var}[\epsilon_t] = \sigma^2$.

The evolution of the intercept term α_t should capture both drifts and shifts in the process. The model that will be used is

$$\alpha_t = \alpha_{t-1} + \delta_t + \tau_t$$

The sequence of random variables $\{\delta_t\}$ is supposed to represent drift. In general it will be an iid sequence, with $\mathrm{E}[\delta_t] = d\sigma$ and $\mathrm{var}[\delta_t] = (r\sigma)^2$. The special case of $r = 0$ would capture completely deterministic drift of d between successive runs, and $d = 0$ would capture unpredictable, or random, drift. The sequence $\{\tau_t\}$ represents the effects of shifts. The distribution of τ_t should have an impulse at zero, since we postulate that steps do not occur at every run. If a step does occur it will either increase or decrease the intercept term. Therefore, a plausible distribution for τ_t is

$$f_{\tau_t}(x) = p_0 u_1(0) + p_1 \phi(m_1, s_1) + p_2 \phi(m_2, s_2)$$

where $p_0 + p_1 + p_2 = 1$, $m_1 > 0$, $m_2 < 0$, $\phi(m, s)$ is the normal density function with mean m and standard deviation s, and $u_1(x)$ is the unit impulse function at location x. In other words, a three-sided coin is flipped and depending on the outcome, the step-size is zero, probably positive, or probably negative.

The next two chapters will detail the steps of the gradual and rapid modes of the current version of the algorithm and discuss some enhancements.

Chapter 4

Gradual Mode

As long as the process is operating in accordance with the process model, the algorithm will be in gradual mode. In this mode, two questions need to be answered after each run of the process:

1. How should the process model be updated ?

2. What should the next recipe for the process be ?

These questions will be discussed in this chapter. First, the procedure used to update the process model will be described.

4.1 Estimating the Process Level

As suggested already, the purpose of the gradual mode is to compensate for drift in the process. The drift could be caused by the aging of components, or by drift in the controllers which maintain the setpoints of such inputs as gas flow rates. We will introduce the algorithm with example. Suppose that we have the following simple single input, single output process: At each run, the process output y_t is linearly related to the process input x_t, and the process is not corrupted by noise. In other words, the process is completely specified by its intercept and slope parameters, or symbolically, $y_t = \alpha_t + \beta x_t$. This situation is depicted in figure 4-1. However, the process parameters are unknown to us. Instead we have the estimated intercept term a_{t-1} and the estimated slope b. We assume in this example that the slope does not

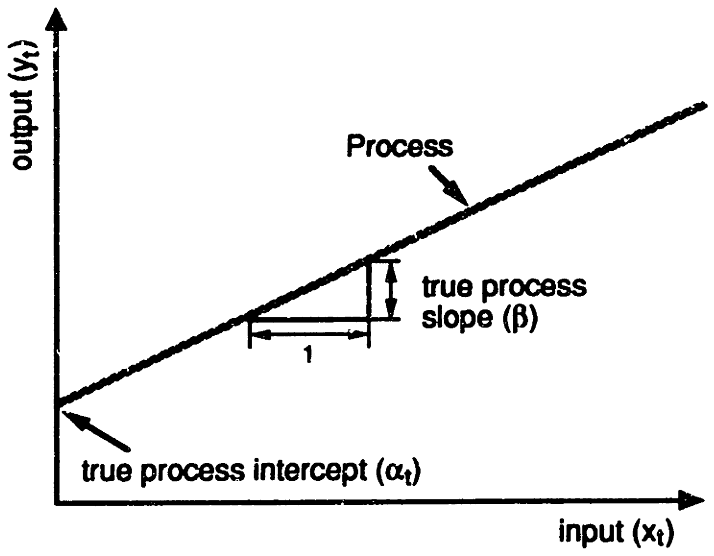

output (y_t)
Process
true process slope (β)
1
true process intercept (α_t)
input (x_t)

Figure 4-1: This figure shows a hypothetical one input, one output process, represented by the shaded line. The process is perfectly linear and is not corrupted by noise. Therefore, the process is characterized by its intercept and slope parameters, which are shown on the figure. For simplicity, the process slope is assumed not to change from one run to the next, but the process intercept might.

change, and is estimated before we start controlling the process. The intercept term, on the other hand, might change with time, and we try keep up with this change by revising the estimated intercept whenever a new output measurement becomes available. Before run t, measurements from runs 1 to $t-1$ are available, and the intercept estimate based on this information is called a_{t-1}. The true process and our process model are shown in figure 4-2. In terms of this figure, the process drift can be visualized by letting the shaded line, representing the process, change its vertical position slightly between successive runs.

With this background, the control strategy can be stated. We use our current model of the process to predict what the output for the next run would be, as a function of the input for that run, and select an input value for which the predicted output is on target. This is illustrated in figure 4-3. Since our estimates are not perfect, the recipe chosen by the controller may not be the ideal one, and the actual output value may be different from the predicted one. But if the slope estimate is "good enough" and the intercept estimate is updated in an intelligent way, the input

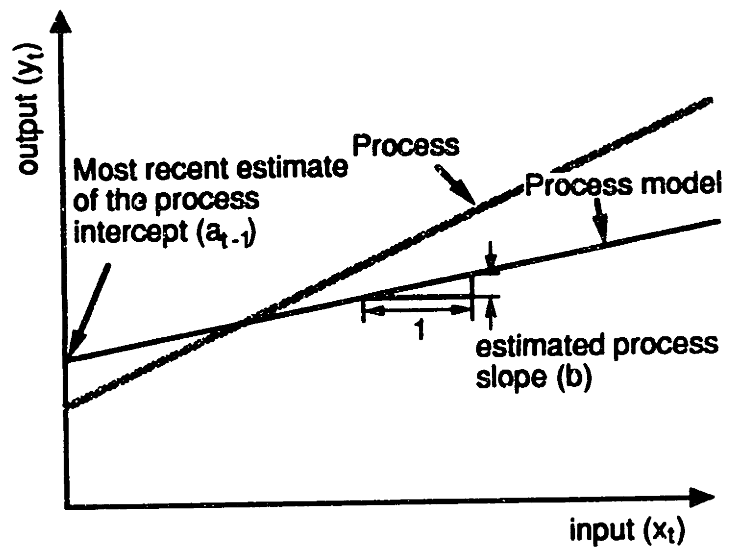

output (yt)
Most recent estimate of the process intercept (at-1)
Process
Process model
1
estimated process slope (b)
input (xt)

Figure 4-2: Here, both the true process (the shaded line) and a model of the process (the solid line) are shown. The parameters of the true process are unknown to use, but they can be estimated. The slope parameter is estimated beforehand, but the estimate of the intercept is updated whenever a new measurement becomes available. The most recent estimate of the intercept term available before run t is the one which is based on information from runs 1 through $t-1$, and is called a_{t-1}.

settings will converge to the ideal recipe and the output values will converge to target.

By now, the only missing piece of the algorithm is the procedure for updating the constant term. Suppose that data from only one run was available, i.e. we knew the values of x_1 and y_1. An obvious method of estimating the intercept term would be to solve the equation $y_1 = a_1 + bx_1$ for a_1. The solution is $a_1 = y_1 - bx_1$. More generally, when data from t runs is available, it is reasonable to compute the estimate at as some weighted average of the numbers $(y_1 - bx_1), (y_2 - bx_2), \ldots, (y_t - bx_t)$. Since we want the estimate to represent the current state of the process, the average should put the most weight on the most recent data. A simple way of doing this is to use an n-point uniform moving average, i.e.

$$a_t = \sum_{i=t-n+1}^{t} \left(\frac{1}{n}\right)(y_i - bx_i)$$

But this scheme seems has the flaw that it puts a weight of $(1/n)$ on a point which is

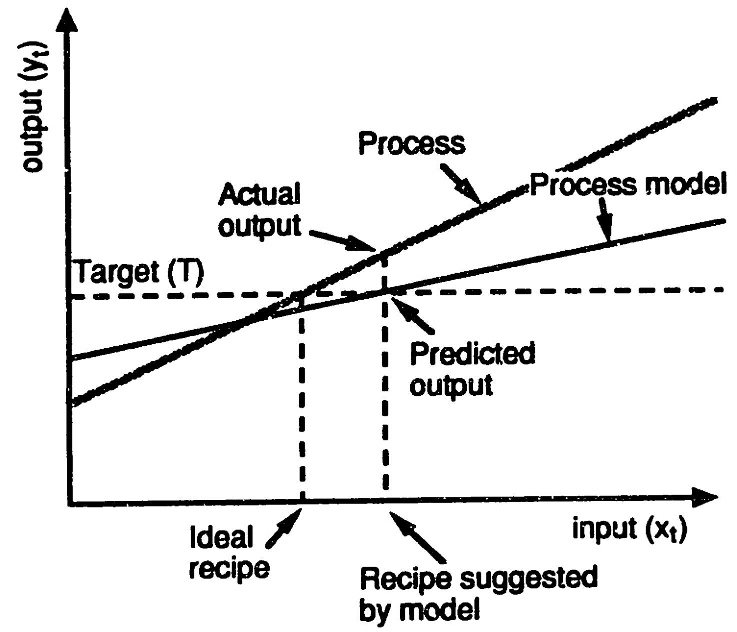


Figure 4-3: Shown on this figure are the true process, the process model, and a horizontal dashed line which intercepts the output axis at the target value T for the output. The intersection of this line with the true process defines where the ideal recipe would be. But the recipe suggested by the process model is determined by the intersection of the dashed line and the process model. The predicted output value using this recipe is on target, but the actual output value will be determined by the true process, as shown.

n runs old, but a weight of zero on a point which is $n+1$ runs old. A more desirable weighing scheme would let the weight of a point decay gradually with age, since there is no reason to believe that the information conveyed by a measurement suddenly becomes worthless, rather its value decreases gradually with age. The Exponentially Weighted Moving Average (EWMA) accomplishes this. It is defined as follows:

$$a_t = \sum_{i=1}^{t} w(1-w)^{t-i}(y_i - bx_i) + (1-w)^t a_0$$

The weights assigned to the data by the EWMA decay geometrically with age. Figure 4-4 contrasts these weights with those assigned by the uniform moving average. The EWMA parameter w is the weight assigned to the most recent data point. This can be seen by expressing the EWMA in the recursive form $a_t = w(y_t - bx_t) + (1-w)a_{t-1}$.

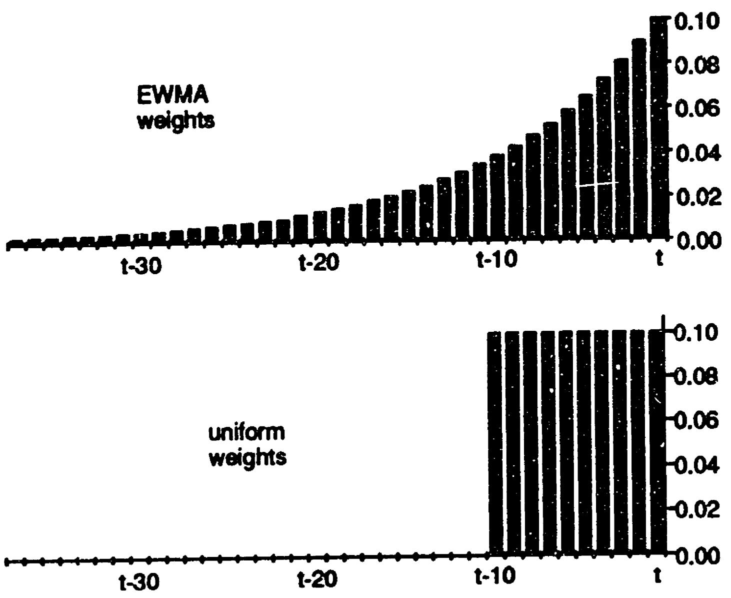


Figure 4-4: A uniform moving average puts equal weights on the last n measurements (in this case $n = 10$). An Exponentially Weighted Moving Average (EWMA) put geometrically decaying weights on all the available measurements. The weight put on a measurement which is i runs old (the $(t-i)$th measurement is $w(1-w)^i$, where $0 < w < 1$ (in this case $w = 0.1$).

We will now justify the intuitive arguments made based on the process model postulated in the last section. Recall that the process is assumed to be described, at least locally, by the linear model $y_t = \alpha_t + \beta' \mathbf{x}_t + \epsilon_t$. For the purposes of the Run by Run control algorithm, the prediction equation $\hat{y}_t = a_{t-1} + \mathbf{b}'\mathbf{x}_t$ is used to predict the value of the output at time t. The random variable α_t has been replaced by an estimate a_{t-1}, which is based on the data available from the first $t-1$ runs of the process, and the random variable vector β has been replaced by a vector of estimated process sensitivities $\mathbf{b}$.

If it were true that the vector of first-order coefficients β was deterministic and known, i.e. $\beta = \mathbf{b}$, then the process equation could be rewritten as

$$z_t = y_t - \mathbf{b}'\mathbf{x}_t = \alpha_t + \epsilon_t$$

and the prediction equation would take the simple form

$$\hat{z}_t = \hat{y}_t - \mathbf{b}'\mathbf{x}_t = a_t$$

Now it can be seen that under the assumptions made, the task of updating the process model reduces to revising an estimate a_t of the process level. Before describing how this estimation is performed, the reader should notice that the same procedure could be used for a higher order model, for instance the quadratic model

$$y_t = \alpha_t + \beta'\mathbf{x}_t + \mathbf{x}_t'\Gamma\mathbf{x}_t + \epsilon_t$$

could be rewritten as

$$z_t = y_t - \mathbf{b}'\mathbf{x}_t - \mathbf{x}_t'\mathbf{C}\mathbf{x}_t = \alpha_t + \epsilon_t$$

under the assumption that the first and second-order coefficients are constant. In practice, these coefficients will be time-dependent, but one may hope that they change slowly enough, relative to the intercept term α_t, that the change can be ignored.

In terms of the variable z_t, the two proposed estimation schemes can be written as

$$a_t = \sum_{i=t-n+1}^{t} \left(\frac{1}{n}\right) z_t$$

for the moving average, and

$$a_t = \sum_{i=1}^{t} w(1-w)^{t-i} z_i + (1-w)^t a_0$$

for the EWMA. Here $w \in (0,1)$ is the weight given to the most recent measurement, and a_0 is the initial estimate (before run 1) of the process level.

The EWMA scheme is the method used in the implemented version of the algorithm, and this is the version whose stability will be investigated in chapter 6.

While the EWMA scheme is more satisfying than the moving average scheme, it still appears to be a somewhat arbitrary way to estimate the process level. Although a control law based on an EWMA statistic was shown to be optimal in the example in section 1.2.2, the assumptions made in this example were that all the process parameters were known.

An estimation method based on Bayes Law will be described next. This method is more coherent than using the EWMA, in the sense that the fact that the true value of the intercept term α_t is never known exactly is accounted for explicitly. This is done by postulating a prior probability distribution for the value of the intercept before run 1, and then updating this distribution to reflect measurement data, as it becomes available. Note, however, that this method does not take into account the other process parameters, such as the slope β, the variance of noise σ, and the variance of drift $(r\sigma)^2$. These parameters are still treated as if they were known by the method to be described.

4.1.1 Bayesian updating

To derive this estimation method, an explicit model of how the process evolves with time is needed. We will use the linear model described in chapter 3, with the following simplifications:

- The vector of sensitivities will be assumed constant and known, i.e. $\beta = \mathbf{b}$, so the process can be described by $z_t = y_t - \mathbf{b}'\mathbf{x}_t = \alpha_t + \epsilon_t$.
- Only drift in the process will be considered, i.e. shifts will not be considered.

In summary, we will concentrate on the following representation of the process:

$$\begin{aligned} z_t &= \alpha_t + \epsilon_t \text{ for } t = 1, 2, \ldots && (4.1) \\ \alpha_t &= \alpha_{t-1} + \delta_t && (4.2) \end{aligned}$$

where, as before, ϵ_t is noise, and δ_T is drift between successive runs. We assume that the knowledge about the level α_t available before run 1 can be summarized by a normal distribution with mean a_0 and variance c_0, i.e.

$$(\alpha_0 | D_0) \sim \mathrm{N}(a_0, c_0) \tag{4.3}$$

D_t is taken to represent all available information about the process up to and including run t. Normally, the information generated by run t will consist of the process recipe and the measured output value, so the set D_t would be updated as

$$D_t = D_{t-1} \cup \{(z_t, \mathbf{x}_t)\}$$

Note that $(z_t, \mathbf{x}_t)$ contains the same information as $(y_t, \mathbf{x}_t)$, but the former will be used, for ease of notation. The iid sequences $\{\epsilon_t\}$ and $\{\delta_t\}$ will in this section assumed to be normally distributed, i.e.

$$\begin{aligned} \epsilon_t &\sim \mathrm{NID}(0, \sigma^2) && (4.4) \\ \delta_t &\sim \mathrm{NID}(0, (r\sigma)^2) && (4.5) \end{aligned}$$

Those two sequences are assumed independent of each other and of the initial distribution specified in 4.3.

Now consider the situation after run $t-1$. Suppose the posterior distribution of α_{t-1} after the output measurement y_{t-1} becomes available is a normal distribution:

$$(\alpha_{t-1}|D_{t-1}) \sim \mathrm{N}(a_{t-1}, c_{t-1}) \tag{4.6}$$

From equations 4.2 and 4.5 it follows that the prior distribution of α_t will be

$$(\alpha_t|D_{t-1}) \sim \mathrm{N}(a_{t-1}, c_{t-1} + (r\sigma)^2)$$

Similarly, 4.1 and 4.4 imply that the one-step-ahead forecast distribution for z_t is

$$(z_t|D_{t-1}) \sim \mathrm{N}(a_{t-1}, c_{t-1} + (1 + r^2)\sigma^2)$$

After the measurement y_t becomes available, the posterior distribution of α_t is given by the following proposition:

Proposition 1 *Given the model 4.1-4.2 and the posterior distribution 4.6, the posterior distribution of α_t is $(\alpha_t|D_t) \sim N(a_t, c_t)$ where*

$$\begin{aligned} a_t &= w_t z_t + (1 - w_t) a_{t-1} && (4.7) \\ c_t &= w_t \sigma^2 && (4.8) \end{aligned}$$

$$w_t = \frac{c_{t-1} + (r\sigma)^2}{c_{t-1} + (1+r^2)\sigma^2}$$

Proof: Let f_{α_t} and f_{z_t} denote the pdf's for α_t and z_t. By Bayes Law,

$$\begin{aligned}
f_{\alpha_t}(a|D_t) &= f_{\alpha_t}(a|D_{t-1} \cup \{(z_t, x_t)\}) \\
&= \frac{f_{z_t}(z|D_{t-1}, \alpha_t = a) f_{\alpha_t}(a|D_{t-1})}{f_{z_t}(z|D_{t-1})} \\
&= k_0 f_{z_t}(z|D_{t-1}, \alpha_t = a) f_{\alpha_t}(a|D_{t-1}) \\
&= k_1 \exp\left\{-\frac{1}{2\sigma^2}(z-a)^2\right\} \exp\left\{-\frac{1}{2(c_{t-1}+(r\sigma)^2)}(a-a_{t-1})^2\right\} \\
&= k_2 \exp\left\{a^2\left(-\frac{1}{2\sigma^2} - \frac{1}{2(c_{t-1}+(r\sigma)^2)}\right) + a\left(\frac{z}{\sigma^2} + \frac{a_{t-1}}{(c_{t-1}+(r\sigma)^2)}\right)\right\} \qquad (4.9)
\end{aligned}$$

Here, k_0, k_1, and k_2 are constants in the sense that they do not depend on a. The second degree polynomial in a in the exponent of 4.9 can be rewritten as:

$$\begin{aligned}
& a^2\left(-\frac{1}{2\sigma^2} - \frac{1}{2(c_{t-1}+(r\sigma)^2)}\right) + a\left(\frac{z_t}{\sigma^2} + \frac{a_{t-1}}{(c_{t-1}+(r\sigma)^2)}\right) \\
&= -\frac{1}{2}\left(\frac{1}{\sigma^2} + \frac{1}{c_{t-1}+(r\sigma)^2}\right)\left\{a^2 - 2a\frac{z_t(c_{t-1}+r\sigma^2) + a_{t-1}\sigma^2}{c_{t-1}+(1+r^2)\sigma^2}\right\} \\
&= -\frac{1}{2c_t}\left\{a^2 - 2aa_t + a_t^2 - a_t^2\right\} \\
&= -\frac{1}{2c_t}\left\{(a-a_t)^2 - a_t^2\right\}
\end{aligned}$$

Substituting this into equation 4.9 results in

$$f_{\alpha_t}(a|D_t) = k_3 \exp\left\{-\frac{1}{2c_t}(a-a_t)^2\right\}$$

where k_3 is a constant normalizing factor, which should be selected to let the density integrate to unity. This implies that the posterior density of α_t is a normal distribution with mean a_t and variance c_t. ■

Since the distribution of α_0 is normal, it can be seen by induction that the preceding argument is valid for all $t = 1, 2, \ldots$.

The form of the updating equation 4.7 is identical to the EWMA updating equation, except that the weight w is time dependent. The following proposition shows that in the limit, the two are equivalent.

Proposition 2 *The sequence $\{w_t\}$ converges to*

$$w = \frac{r}{2}\left(\sqrt{1+4/r}-1\right)$$

Also, the sequence $\{c_t\}$ (the variance of the distribution of a_t) converges to $c = w\sigma^2$.

Proof: Since $c_t = w_t\sigma^2$, it is sufficient to prove that $\{c_t\} \to c$.

Assuming $\infty > \sigma^2 > 0$, $c_{t-1} > 0$, and using $r \geq 0$, it is seen that

$$0 < \frac{c_{t-1} + r\sigma^2}{c_{t-1} + (1+r)\sigma^2} < 1 \Rightarrow 0 < w_t < 1$$

This, combined with the updating equation for c_t, implies

$$0 < c_t < \sigma^2$$

Assuming $c_0 > 0$, this relation holds for all t, by induction. Therefore, $\{c_t\}$ is bounded.

If it could be shown that $\{c_t\}$ was monotone also, then the sequence would have been shown to converge. To this end, consider the sequence c_t^{-1}, which is defined by

$$c_t^{-1} = \frac{c_{t-1} + (1+r)\sigma^2}{c_{t-1} + r\sigma^2}\left(\frac{1}{\sigma^2}\right) = \frac{1}{\sigma^2} + \frac{1}{c_{t-1} + r\sigma^2}$$

After some algebraic manipulation, the following relation can be shown to hold:

$$\begin{aligned} c_t^{-1} - c_{t-1}^{-1} &= \frac{c_{t-2}c_{t-1}}{(c_{t-1} + r\sigma^2)(c_{t-2} + r\sigma^2)} \times \left(c_{t-1}^{-1} - c_{t-2}^{-1}\right) \\ &= K_t\left(c_{t-1}^{-1} - c_{t-2}^{-1}\right) \end{aligned}$$

where $0 < K_t \leq 1$, which implies that $\{c_t^{-1}\}$ is monotone. Therefore, $\{c_t\}$ must be monotone also, and hence must converge.

The updating equation for c_t (4.8) implies that if c_t does converge to c, then c must satisfy

$$c = \frac{c + r\sigma^2}{c + (1+r)\sigma^2}\sigma^2$$

which is a quadratic equation in c. The two solutions are $c = (r/2)(\pm\sqrt{1+4/r}-1)\sigma^2$. Since c_t has already been shown to be bounded between 0 and σ^2, it must converge to the positive root, $c = (r/2)(\sqrt{1+4/r}-1)\sigma^2$. ∎

Since the Bayesian updating scheme has been shown to be asymptotically equivalent to an EWMA scheme, one might ask whether the benefits of using the Bayesian scheme outweigh the cost of added complexity. The answer is that the added complexity is minimal; the updating formulas 4.7 and 4.8 are easy to implement. The major benefit of the Bayesian scheme is that it is open to *intervention*, in the sense that if external information about a change in the level a_t becomes available, this information could be incorporated in a natural way by changing the parameters a_t and c_t of the distribution of α_t. Under the EWMA scheme, one could change the current estimate a_t, but it is not obvious how to account for the increase or decrease in uncertainty, which in the Bayesian scheme would be represented by c_t.

Further generalizations of the Bayesian updating scheme can be found in [WH89]. For example, suppose that the parameter σ was not assumed to be known, but rather that an initial distribution of σ was given. In particular, suppose

$$(1/\sigma^2|D_{-1}) \sim \text{gamma}(n_0/2, d_0/2)$$

i.e. the inverse of the measurement error variance has an initial gamma distribution, with parameters n_0 and d_0. The gamma density is

$$f(t) = \left(\frac{(d_0/2)^{n_0/2}}{\Gamma(n_0/2)}\right) t^{\frac{n_0}{2}-1} e^{-\frac{td_0}{2}}$$

If the system is assumed to evolve in the same way as before (equations 4.1 and 4.2) and the same independence assumptions are made, then the following proposition holds:

Proposition 3 *For all t, the distributions of α_t and σ^2 can be updated as follows:*

Conditional on σ^2, $(\alpha_t|D_t) \sim N(a_t, \sigma^2 c_t^*)$ where

$$\begin{aligned} a_t &= \frac{c_{t-1}^* + r}{c_{t-1}^* + 1 + r} z_t + \frac{1}{c_{t-1}^* + 1 + r} a_{t-1} \\ c_t^* &= \frac{c_{t-1}^* + r}{c_{t-1}^* + 1 + r} \end{aligned}$$

Conditional on D_t, the inverse of the measurement variance, $1/\sigma^2$, has the gamma distribution with parameters $n_t = n_{t-1} + 1$ and $d_t = d_{t-1} + (z_t - a_{t-1})^2/(c_{t-1}^* + 1 + r)$.

Unconditional on σ^2, α_t has the t-distribution, with expected value a_t and variance $c_t = (d_t/n_t)c_t^*$.

For a proof of this proposition, see [WH89].

4.2 Selecting New Recipe

Once the process model has been updated, a new recipe, $\mathbf{x}_{t+1}$ will be selected, which satisfies

$$\hat{y}_{t+1} = a_t + \mathbf{b}'\mathbf{x}_{t+1} = T$$

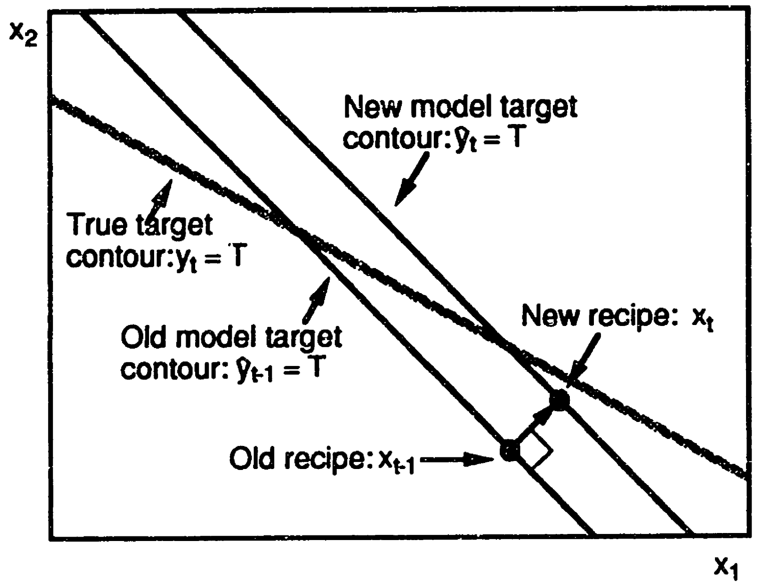
Since this equation has infinitely many solutions (assuming $\mathbf{b} \neq 0$), the choice of $\mathbf{x}_{t+1}$ has to be constrained further. We choose to select $\mathbf{x}_{t+1}$ so as to minimize the distance $||\mathbf{x}_{t+1} - \mathbf{x}_t||$, because unnecessary changes in the recipe are undesirable, and because the process model may only be valid locally. This is illustrated in figure 4-5, where all points on the "new model contour" satisfy the equation $T = a_{t-1} + \mathbf{b}'x_t$.

Therefore, $\mathbf{x}_{t+1}$ should be chosen as a solution to the non-linear program

$$\begin{aligned} \min \quad & ||\mathbf{x}_{t+1} - \mathbf{x}_t||^2 \\ \text{subject to} \quad & \mathbf{b}'\mathbf{x}_{t+1} = T - a_t \end{aligned} \tag{4.10}$$

Since the constraint set and objective function are both convex, the Kuhn-Tucker conditions ([Lue84]) are both necessary and sufficient to guarantee optimality, under the usual regularity conditions. For this program, the conditions are:

$$\nabla f(\mathbf{x}_{t+1}) + \lambda \nabla g(\mathbf{x}_{t+1}) = 0$$



x_2
x_1
New model target contour: $\hat{y}_t = T$
True target contour: $y_t = T$
Old model target contour: $\hat{y}_{t-1} = T$
New recipe: x_t
Old recipe: x_{t-1}

Figure 4-5: This figure illustrates the gradual mode algorithm for a two input process. The vertical and horizontal axes represent the two inputs. The shaded line is the contour along which the process output is on target (assuming there is no noise in the process). At run $t-1$, the process is run at the "old recipe" shown. Since this recipe is not on the target contour, the output measurement will be different from the target. Suppose the measured value is larger than the target value. In response, the process model (which is a plane in 3 dimensional space) is shifted down (in the output coordinate), to compensate partially for the deviation from target. Correspondingly, the contour along which the model predicts that the output will be on target shifts as shown in the picture. The new recipe is chosen as that point on the new model contour which is closest to the old recipe, and is obtained by projecting the old recipe onto the new model contour.

$$g(\mathbf{x}_{t+1}) = 0$$

where $f(\mathbf{x}_{t+1}) = ||\mathbf{x}_{t+1} - \mathbf{x}_t||^2$ and $g(\mathbf{x}) = \mathbf{b}'\mathbf{x}_{t+1} - T + a_t$. After taking the appropriate derivatives, these conditions can be written as

$$2(\mathbf{x}_{t+1} - \mathbf{x}_t) + \lambda \mathbf{b} = 0 \quad (4.11)$$

$$\mathbf{b}'\mathbf{x}_{t+1} - T + a_t = 0 \quad (4.12)$$

These equations can be solved for λ, by pre-multiplying 4.11 with $\mathbf{b}'$, solving for

$\mathbf{b}'\mathbf{x}_{t+1}$ and substituting into 4.12, to get

$$\lambda = \frac{2}{||\mathbf{b}||^2}(\mathbf{b}'\mathbf{x}_t - T + a_t)$$

This expression for λ can then be substituted into 4.11. Solving for $\mathbf{x}_{t+1}$,

$$\mathbf{x}_{t+1} = \frac{T - a_t}{||\mathbf{b}||^2}\mathbf{b} + \left(\mathbf{I} - \frac{\mathbf{b}\mathbf{b}'}{||\mathbf{b}||^2}\right)\mathbf{x}_t$$

This is the currently implemented method of selecting the next recipe. Next, methods of selecting the next recipe when the process model is second order will be described.

4.2.1 Second-order Process Model

If the process model is a second-order polynomial, the constraints 4.10 should be replaced with

$$\mathbf{x}'_{t+1}\mathbf{C}\mathbf{x}_{t+1} + \mathbf{b}'\mathbf{x}_{t+1} = T - a_t$$

The solution set of this system is not convex (it is a conic section), therefore the Kuhn-Tucker conditions are necessary but not sufficient for optimality.

We will now describe two approaches to finding the optimal value of the above non-linear program. The first approach transforms the vectors $\mathbf{x}_t$ into a vector space in which the quadratic constraints can be expressed as $\mathbf{u}'_t\mathbf{u}_t = r^2$. The second approach is approximate, and consists of following the gradient of the estimated process function from the point $\mathbf{x}_t$ until the constraint is satisfied.

Transformation of Variables Suppose $\mathbf{C}$ is positive definite. Then $\mathbf{C}^{1/2}$ exists, is unique, and is invertible. Hence, the following transformation is invertible:

$$\mathbf{u}_{t+1} = \mathbf{C}^{\frac{1}{2}}\mathbf{x}_{t+1} + \frac{1}{2}\mathbf{C}^{-\frac{1}{2}}\mathbf{b}$$

In terms of $\mathbf{u}_{t+1}$, the non-linear program can be written as

$$\begin{array}{rl} \min & (\mathbf{u}_{t+1} - \mathbf{u}_t)'\mathbf{C}^{-1}(\mathbf{u}_{t+1} - \mathbf{u}_t) \\ \text{subject to} & \mathbf{u}'_{t+1}\mathbf{u}_{t+1} = r^2 \end{array}$$

where $r^2 = \frac{1}{4}\mathbf{b}'\mathbf{C}^{-1}\mathbf{b} - a_t + T$, which will be assumed to be a positive quantity. The Kuhn-Tucker conditions in terms of $\mathbf{u}_{t+1}$ are

$$\begin{aligned} \mathbf{C}^{-1}(\mathbf{u}_{t+1} - \mathbf{u}_t) + 2\lambda\mathbf{u}_{t+1} &= 0 & (4.13) \\ \mathbf{u}'_{t+1}\mathbf{u}_{t+1} &= r^2 & (4.14) \end{aligned}$$

Equation 4.13 can be rewritten as

$$\mathbf{u}_{t+1} = \left(2\lambda\mathbf{I} + \mathbf{C}^{-1}\right)^{-1}\mathbf{C}^{-1}\mathbf{u}_t \qquad (4.15)$$

Suppose $\mathbf{C}$ has distinct eigenvalues d_i and corresponding normalized eigenvectors $\mathbf{m}_i$, $i = 1, \ldots, n$, where $||\mathbf{m}_i|| = 1$. Then $\mathbf{C}$ can be written as $\mathbf{C} = \mathbf{MDM}^{-1}$, where $\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_n]$ and $\mathbf{D} = \text{diag}(d_1, \ldots, d_n)$. It can then be verified that

$$\begin{aligned} \left(2\lambda\mathbf{I} + \mathbf{C}^{-1}\right)^{-1} &= \mathbf{M}\left(2\lambda\mathbf{I} + \mathbf{D}^{-1}\right)^{-1}\mathbf{M}^{-1} \\ &= \mathbf{M}\left[\text{diag}\left(\frac{d_i}{2\lambda d_i + 1}\right)\right]\mathbf{M}_{-1} \end{aligned}$$

From equation 4.15 it then follows that

$$\begin{aligned} \mathbf{u}_{t+1} &= \left(2\lambda\mathbf{I} + \mathbf{C}^{-1}\right)^{-1}\mathbf{C}^{-1}\mathbf{u}_t \\ &= \mathbf{M}\left[\text{diag}\left(\frac{1}{2\lambda d_i + 1}\right)\right]\mathbf{M}^{-1}\mathbf{u}_t & (4.16) \end{aligned}$$

Let $\mathbf{k} = \mathbf{M}^{-1}\mathbf{u}_t$ and $\mathbf{l} = \text{diag}\left(\frac{1}{2\lambda d_i + 1}\right)\mathbf{k}$. Then we can write $\mathbf{u}_{t+1} = \mathbf{Ml}$. Substituting 4.16 into 4.14 we get

$$\begin{aligned} r^2 &= \mathbf{l}'\mathbf{M}'\mathbf{Ml} \\ &= \left(\sum_{i=1}^{n} l_i\mathbf{m}'_i\right)\left(\sum_{i=1}^{n} l_i\mathbf{m}_i\right) \\ &= \sum_{i=1}^{n}(l_i)^2 + \sum_{i \neq j} l_i l_j \mathbf{m}'_i\mathbf{m}_j & (4.17) \end{aligned}$$

By observing that $l_i = \frac{k_i}{2\lambda d_j+1}$ and cross-multiplying equation 4.17 the following polynomial of degree $2n$ in λ is obtained:

$$\begin{aligned} r^2 \prod_{i=1}^{n} (2\lambda d_i + 1)^2 &= \sum_{i=1}^{n} k_i^2 \prod_{i \neq j} (2\lambda d_j + 1)^2 \\ &+ 2 \sum_{i \neq j} k_i k_j \mathbf{m}_i' \mathbf{m}_j \prod_{r \neq i, r \neq j} (2\lambda d_r + 1)^2 (2\lambda d_i + 1)(2\lambda d_j + 1) \end{aligned}$$

As a result, the problem boils down to finding that root of the preceding polynomial which minimizes $||\mathbf{x}_{t+1} - \mathbf{x}_t||$. Since this can be a difficult problem when the number of input variables n is large, a simpler but approximate approach will be described next.

Steepest descent approach The second approach to finding the next recipe is simpler to implement, but is approximate in the sense that it is not guaranteed to minimize the distance $||\mathbf{x}_{t+1} - \mathbf{x}_t||$. It consists of following the gradient of the estimated process function $\mathbf{x}'\mathbf{C}\mathbf{x} + \mathbf{b}'\mathbf{x} + a_t$ from $\mathbf{x}_t$, until the value of the estimated process function equals the target value T. In other words, $\mathbf{x}_{t+1}$ is selected as

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha(\mathbf{C}\mathbf{x}_t + \mathbf{b})$$

where α satisfies $A\alpha^2 + B\alpha + C = 0$, with

$$\begin{aligned} A &= \mathbf{x}_t'\mathbf{C}^3\mathbf{x}_t + \mathbf{b}'\mathbf{C}\mathbf{b} + 2\mathbf{x}_t'\mathbf{C}^2\mathbf{b} \\ B &= 2\mathbf{x}_t'\mathbf{C}^2\mathbf{x}_t + 3\mathbf{x}_t'\mathbf{C}\mathbf{b} + \mathbf{b}'\mathbf{b} \\ C &= \mathbf{x}_t'\mathbf{C}\mathbf{x}_t + \mathbf{b}'\mathbf{x}_t - T + a_t \end{aligned}$$

Since $|\alpha|$ is proportional to the distance $||\mathbf{x}_{t+1} - \mathbf{x}_t||$, one should select the root with the smaller absolute value. If the roots are complex, then this approach does not work. Figure 4-6 illustrates this method, and the situation when the roots are complex. Possible enhancements to the steepest descent approach, to remedy the problem illustrated in figure 4-6, are

- Scaling the direction of descent by the Hessian, i.e. letting $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha\mathbf{C}^{-1}(\mathbf{C}\mathbf{x}_t + \mathbf{b})$, as in Newton's method ([Lue84]).

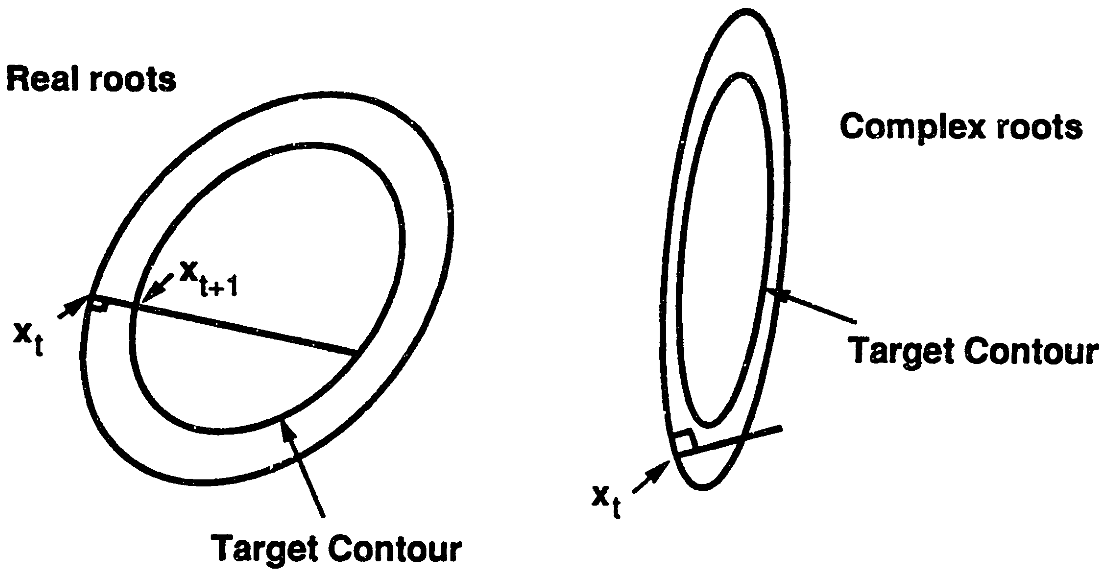


Figure 4-6: Approximate projection onto quadratic contour. A line orthogonal to the function $\mathbf{b}'\mathbf{x} + \mathbf{x}'\mathbf{C}\mathbf{x}$ at the current recipe $\mathbf{x}_t$ is extended until it intersects the target contour $a_t + \mathbf{b}'\mathbf{x} + \mathbf{x}'\mathbf{C}\mathbf{x} = T$. However, if the quadratic equation discussed in the text has complex roots, then the line will never intersect the target contour, as shown on the right.

- Applying the steepest descent approach iteratively, until the target contour is reached.

Dependency on scaling A deficiency of this approach to selecting the next recipe is that the distance $||\mathbf{x}_{t+1} - \mathbf{x}_t||$ depends on the units which are used to measure the components of the recipe. For example, suppose $\mathbf{x}$ was two dimensional, the two components being a flow rate, measured in sccm, and an injector position, measured either in cm or m. Suppose the last recipe was $\mathbf{x}_t' = (100 \text{ sccm}, 100 \text{ cm}) = (100 \text{ sccm}, 1 \text{ m})$, the vector of estimated first-order coefficients was $\mathbf{b}' = (1 \text{ Å/ sccm}, 1 \text{ Å/ cm}) = (1 \text{ Å/ sccm}, 100 \text{ Å/ m})$, and the target was $T = 210\text{Å}$. Then the next recipe, $\mathbf{x}_{t+1}$ would have to satisfy

$$(\text{flow rate in sccm}) + (\text{injector position in cm}) = 210$$

Two solutions are

$$\begin{aligned}\bar{\mathbf{x}}_{t+1} &= \begin{pmatrix} 100 \text{ sccm} \\ 110 \text{ cm} \end{pmatrix} \\ \bar{\bar{\mathbf{x}}}_{t+1} &= \begin{pmatrix} 110 \text{ sccm} \\ 100 \text{ cm} \end{pmatrix}\end{aligned}$$

If injector position is measured in cm, then

$$\begin{aligned}||\bar{\mathbf{x}}_{t+1} - \mathbf{x}_t|| &= 10 \\ ||\bar{\bar{\mathbf{x}}}_{t+1} - \mathbf{x}_t|| &= 10\end{aligned}$$

If injector position is measured in m,

$$\begin{aligned}||\bar{\mathbf{x}}_{t+1} - \mathbf{x}_t|| &= 0.1 \\ ||\bar{\bar{\mathbf{x}}}_{t+1} - \mathbf{x}_t|| &= 10\end{aligned}$$

To get around this difficulty, the user of the Run-by-Run Controller Algorithm is asked to provide estimated upper and lower bounds for each control variable, say l_i and u_i for each component x_i of $\mathbf{x}$. Then, x_i is transformed into

$$x_i' = \frac{x_i - (u_i + l_i)/2}{(u_i - l_i)/2}$$

The transformed coordinates x_i' are used to compute the new recipe, which is then transformed back to the original units, using

$$x_i = \frac{1}{2}((u_i - l_i)x_i' + (u_i + l_i))$$

Chapter 5

Rapid Mode

When generalized SPC signals an alarm, the RbR controller enters rapid mode, whose function is to rapidly compensate for a possible major disturbance to the process. In order for this compensation to be of the appropriate magnitude, two questions need to be answered:

1. Was the alarm really caused by a disturbance, or was it a false alarm, caused by random fluctuation ?

2. If a disturbance did occur, what was its nature ?

The goal of rapid mode is to make good use of the data to answer those two questions. The strategy employed is the following: First, it will be assumed that a disturbance did occur, and that it had the form of a step function. Least squares estimates of the magnitude and location of the step will then be computed. These estimates provide a tentative answer to the second question. Next, the degree of certainty that a disturbance of the estimated form did indeed occur will be assessed. This will be done using concepts from Bayesian statistics. The result will be the posterior probability that the alarm was not a false alarm, which provides an answer to the first question. Of course, the actual disturbance might not have the exact form of a step disturbance. It might have various forms, but the most significant feature is that it is likely to have changed the process level by a "large" amount (since it caused an alarm) in a "small" number of runs (since gradual mode was not able to compensate for it). In particular, we expect the change in process level to be on the order of 3σ

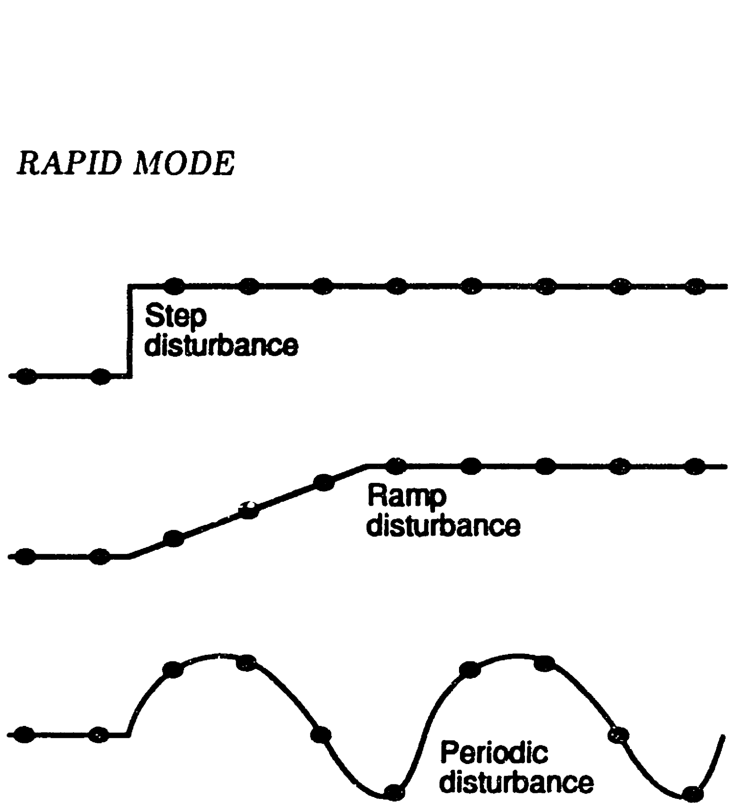


Figure 5-1: Three important categories of disturbances to the mean level of a process are steps, ramps, and periodic disturbances. The first two of these can be approximated by a step, but the third cannot.

in less than 10 runs. Important categories of disturbances that have this feature are steps, steep ramps, and the beginning of a periodic disturbance, as shown in figure 5-1 (note that in illustrating the form of the disturbances, we are ignoring drift and noise in the process for a moment). We will try to approximate the actual disturbance by a step, no matter what its form. This seems reasonable for a step and a steep ramp, but for a periodic disturbance, this might cause problems.

5.1 Estimation of Step Location and Magnitude

The procedure used to estimate the parameters of the step function is illustrated in figure 5-2. The data points that are shown in this figure are the values of $z_t = y_t - \mathbf{b}'\mathbf{x}_t$, i.e. the output measurements, adjusted for the effect of the input variables, for the last few runs. Two horizontal lines are fitted to this data, as shown in the figure, to minimize the sum of squared deviations of the data from the lines. The position of the breakpoint and the levels of the two lines are varied to find the combination that minimizes the sum of squared deviations. The position of the breakpoint provides an

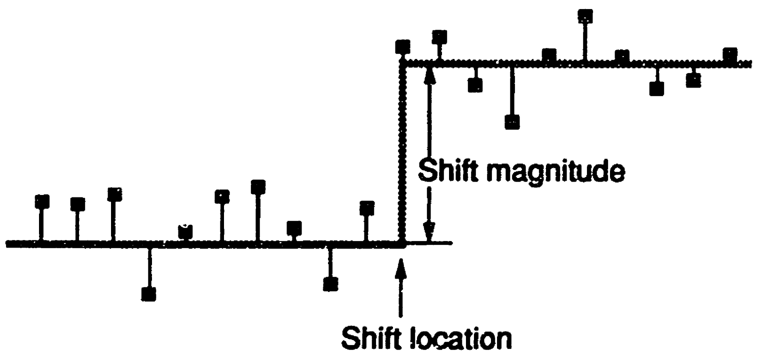


Figure 5-2: When a generalized SPC alarm occurs, the RbR controller attempts to identify a shift in the intercept term during the last few runs. A shift has two parameters, its magnitude and location. These parameters are estimated using a least squares procedure.

estimate of the location (in time) of the shift disturbance, and the distance between the lines is an estimate of its magnitude. It should be noted that this procedure implicitly assumes that the change in process level because of drift during the last few runs is small enough (compared to the magnitude of the step) to be ignored.

5.1.1 Formal Description of the Estimation Procedure

In estimating the magnitude and location of the shift, the algorithm uses data from the last k runs. As before, let $z_t = y_t - \mathbf{b}'\mathbf{x}_t$. Assuming that gradual mode is engaged, the identity $T = a_{t-1} + \mathbf{b}'\mathbf{x}_t$ holds for all t, so z_t can be written as

$$\begin{aligned} z_t &= y_t - T + a_{t-1} \\ &= a_{t-1} + e_t \end{aligned}$$

where e_t is the output deviation from target, or $y_t - T$. In other words, z_t equals the sum of the most recent estimate of the intercept term and the control error e_t. We will now introduce the quantity $\hat{z}_i$, which may be thought of as an estimator for the intercept term α_{i+1}. Note that this intercept term estimator may take on values different from the EWMA estimator a_i. In particular, the estimator $\hat{z}_i$ will be restricted to the following form:

$$\hat{z}_i = \begin{cases} a_- & \text{for } i = t-k+1, \ldots, t-m \\ a_+ & \text{for } i = t-m+1, \ldots, t \end{cases}$$

The interpretation of this is that the estimated intercept before the shift is a_-, the estimated intercept after the shift is a_+, and the shift was estimated to have occurred between runs $t-m$ and $t-m+1$. The parameters a_-, a_+, and m are chosen to minimize the sum of squared deviations

$$SS(a_-, a_+, m) = \sum_{i=t-k+1}^{t} (z_i - \hat{z}_i)^2$$

If the value of m is fixed, then this becomes a standard least squares regression problem. The minimizing values are found by repeating this regression for every value of m in the range $0, \ldots, k-1$, and selecting that value of m which minimizes the sum of squared deviations.

Rationale for the Estimation Procedure The rationale for this estimation scheme is the following: Suppose that the process drift is slow enough that the estimated intercept term a_t does does not change much during k runs, so we can write

$$z_i \simeq a + e_i \text{ for } i = t-k+1, \ldots, t$$

for some constant a. Suppose also that a step of size τ really did occur, between runs $t-m$ and $t-m+1$, and that the gradual mode responds very slowly, so the following is a good approximation:

$$z_i \simeq \begin{cases} a + \epsilon_i & \text{for } i = t-k+1, \ldots, t-m \\ a + \tau + \epsilon_i & \text{for } i = t-m+1, \ldots, t \end{cases}$$

where ϵ_i is a random error term with $\mathrm{E}[\epsilon_i] = 0$ and $\mathrm{var}[\epsilon_i] = \sigma^2$. If all this were true, then the estimation procedure described would be appropriate.

A Potential Enhancement One might be able to design a better procedure by explicitly taking into account that the process is being controlled by the gradual mode algorithm. Suppose that the gradual mode algorithm is using a good estimate of the process sensitivity and an appropriate value for the EWMA weight w. Then, in the absence of steps, the control errors e_i should be approximately distributed as the uncorrelated noise terms ϵ_i. Just after a step, however, the expected value of the control error should be approximately equal to the magnitude of the step and should then decay geometrically until generalized SPC signals an alarm, causing rapid mode to be engaged. The ratio of decay is $1-w$, because the gradual mode compensates for a fraction w of the control error at each run. Based on these informal considerations, an enhanced estimation scheme would be to minimize the sum of squared deviations, as previously defined, subject to the restriction that

$$\hat{z}_i = \begin{cases} a_{i-1} & \text{for } i = t-k+1,\ldots,t-m \\ a_{i-1} + (\Delta a)(1-w)^{i-t+m-1} & \text{for } i = t-m+1,\ldots,t \end{cases}$$

where Δa is an estimate of the magnitude of the shift. This minimization could be performed as the one described previously, by fixing the shift location defined by m, finding the minimizing shift magnitude estimate Δa, repeating this for every value of m from 0 to $k-1$, and choosing the minimizing value of m.

5.2 Assessing the Certainty of Shift

Now that estimates of the parameters of the disturbance are available, we turn to the question of whether a disturbance did indeed occur. Suppose that on any given run t, there is a small probability p that the process level will shift significantly. Then Bayes' rule states that the "posterior probability" that a shift did occur at run t, given data from the next few runs after t is

$$\begin{aligned} &\Pr\{\text{shift at run } t | \text{ data from runs } t+1,\ldots,t+m\} \\ &= \frac{\Pr\{\text{data}|\text{shift}\}p}{\Pr\{\text{data}|\text{shift}\}p + \Pr\{\text{data}|\text{no shift}\}(1-p)} \end{aligned}$$

This updating of the probability of shift can be done sequentially, for one data point at a time, as shown in figure 5-3.

We now have available an estimate of the shift magnitude, say Δa, and the prob-

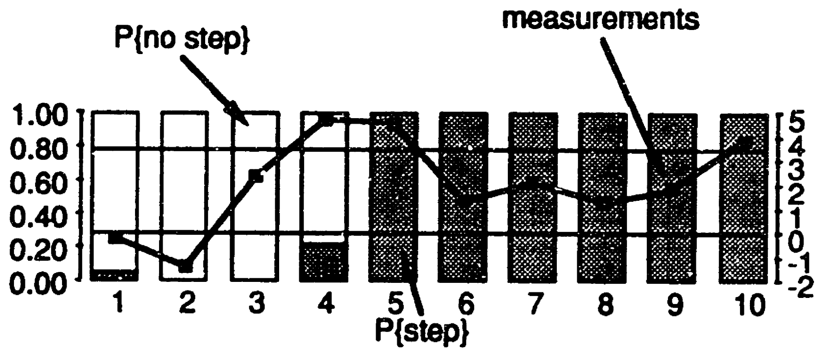


Figure 5-3: This figure shows a simulated step disturbance. The shift occurred between runs 2 and 3. and had a magnitude of 3 σ. The shift was estimated to have occurred between runs 1 and 2, with a magnitude of 3.5 σ. The two horizontal lines represent the estimated level of the process assuming that a shift did occur (the upper line) or did not occur (the lower line), and the black squares represent the data from the process (the one step ahead prediction errors). After run 1, the a priori probability of a shift of magnitude 3.5 σ having occurred was set at 0.05. The vertical columns show how the probability of shift is updated using data from the process. The shaded area represents the probability that a shift occurred, and the non-shaded area is proportional to the probability that it did not. Here, the probability of shift is equal to 1.0 after run 5.

ability that a shift really occurred, say $\bar{p}$. The expected magnitude of shift is then $\bar{p}\Delta a+(1-\bar{p})0 = \bar{p}\Delta a$, and this is the amount by which the intercept term is adjusted. This is shown in figure 5-4, which illustrates how the probability of shift $\bar{p}$ can be interpreted as a "compensation factor".

5.2.1 The Bayesian Updating Procedure

The Bayesian updating procedure for computing the probability that a shift occurred will now be described in more detail. We have available the estimates Δa, of the shift magnitude, and m of how many runs ago the shift occurred. Now define a random variable τ to be the magnitude of the shift, and suppose (somewhat arbitrarily) that conditional on the data available up to and including run $t-m$ (the run immediately before the shift was estimated to have occurred), τ has the following probability mass

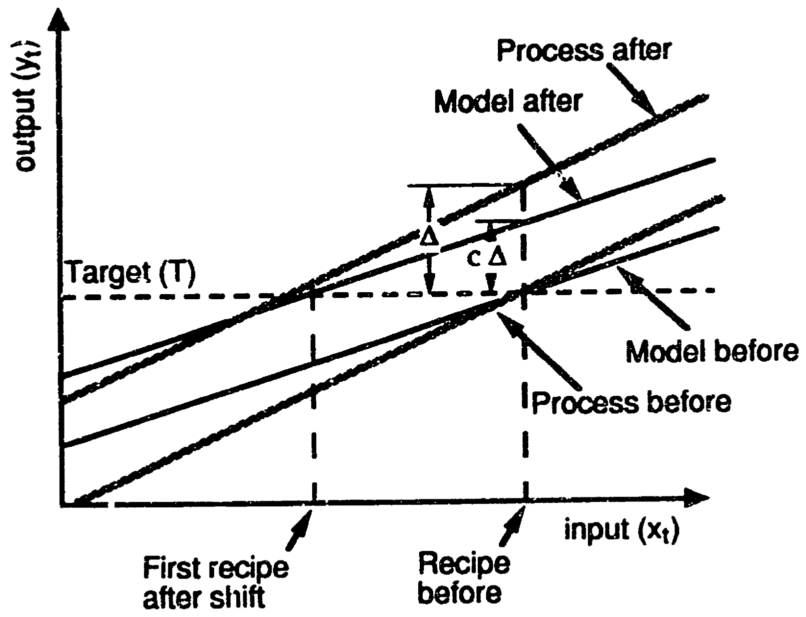


Figure 5-4: After a shift has been estimated to have magnitude Δ, the intercept term of the process model is changed by $\bar{p}\Delta$, where $\bar{p}$ is the probability that the shift really occurred.

function:

$$p_\tau(u|D_{t-m}) = \begin{cases} p & \text{for } u = \Delta a \\ 1-p & \text{for } u = 0 \\ 0 & \text{otherwise} \end{cases}$$

where D_{t-m} is the set of data available after run $t-m$. The value p is some small probability, say 0.05, which is assigned beforehand, as the "a priori probability that a shift of significant magnitude will occur on any given run".

Now recall the argument made to justify the shift estimation procedure. It was argued that if gradual mode responded slowly enough, the quantity z_i was approximately equal to $a + \tau + \epsilon_i$, for runs after the shift occurred. Now suppose that ϵ_i is normally distributed with mean 0 and variance σ^2. Then z_i is approximately normally distributed, with mean $a + \tau$ and variance σ^2. Therefore, the probability mass function for τ, conditional on all the data available after run t, can be computed using Bayes' law:

$$p_\tau(u|D_t) = \frac{\Pr\{z_{t-m+1}, \ldots, z_t|\tau = u\}p_\tau(u|D_{t-m})}{\sum_u \Pr\{z_{t-m+1}, \ldots, z_t|\tau = u\}p_\tau(u|D_{t-m})}$$

In particular, the posterior probability of shift, $\bar{p}$, can be computed as

$$\bar{p} = \frac{\phi_+(z_{t-m+1},\ldots,z_t)p}{\phi_+(z_{t-m+1},\ldots,z_t)p + \phi_-(z_{t-m+1},\ldots,z_t)(1-p)}$$

where $\phi_+(\cdot)$ is a multivariate normal pdf, whose components are uncorrelated, with mean $a + \Delta a$ and variance σ^2, and $\phi_-(\cdot)$ is the same multivariate normal, except that the components have mean a. To perform the computation, it may be more efficient to update the probability of shift sequentially, using the values $z_{t-m+1},\ldots,z_t$ one at a time.

It has already been stated that the intercept term will be changed by the amount $\bar{p}\Delta a$, the expected value of the shift magnitude. To justify this, suppose we wanted to adjust the intercept term by an amount $c\Delta a$, where c is some constant, to minimize the expected squared deviation $\mathrm{E}[(y_{t+1} - T)^2|D_t]$ for the next run. If we suppose that the process sensitivities are known ($\mathbf{b} = \beta$), the process intercept before the step was correctly estimated ($\alpha_{t+1} = a + \tau$), and that drift can be ignored, then

$$y_{t+1} = a + \tau + \mathbf{b}'\mathbf{x}_{t+1} + \epsilon_{t+1}$$

Further, the recipe $\mathbf{x}_{t+1}$ is chosen to satisfy

$$T = a + c\Delta a + \mathbf{b}'\mathbf{x}_{t+1}$$

Subtracting this equation from the previous one, we get

$$y_{t+1} - T = \tau - c\Delta a + \epsilon_{t+1}$$

Therefore, using the identity $\mathrm{var}[X] = \mathrm{E}[X^2] - \mathrm{E}[X]^2$, the expected squared deviation can be written as

$$\begin{aligned}
\mathrm{E}[(y_{t+1} - T)^2|D_t] &= \mathrm{var}[y_{t+1} - T|D_t] + (\mathrm{E}[y_{t+1} - T|D_t])^2 \\
&= \mathrm{var}[\tau|D_t] + \sigma^2 + (\mathrm{E}[\tau|D_t] - c\Delta a)^2 \\
&= \mathrm{var}[\tau|D_t] + \sigma^2 + (\bar{p} - c)^2(\Delta a)^2
\end{aligned}$$

It is now easily verified that the mean squared deviation is minimized by setting c

equal to $\bar{p}$.

5.2.2 The Current Algorithm

In this subsection, the current version of the rapid mode algorithm will be explicitly specified. But first, three features of the algorithm that have not been mentioned yet will be described:

- A complication arises if one shift disturbance generates more than one alarm. This could happen, for example, if the first alarm occurs at the first run after the shift occurred. Since there is only one data point to estimate the magnitude of the shift, this estimate may be poor. Therefore, the current recipe may be adjusted by too much or too little. This increases the probability of an alarm in the next few runs. The following rule deals with this difficulty:

```
if an alarm occurred on the current run then
    if the last estimated shift occurred less than $l$ runs ago then
        remove the most recent adjustment to the intercept term.
    end if
end if
```

  This rule is applied before the location and magnitude of the shift are estimated. When an alarm occurs, and the condition of the above rule is not satisfied, then an alarm counter is set to zero, and the adjustments made to the intercept term before that time will never be removed, i.e. are "locked in".

- Assuming that an alarm is signaled soon after a shift occurs, say two to three runs, there will only be two or three measurements available after the shift. This has two implications. First, the estimate Δa of the shift magnitude will be imprecise. Second, the posterior probability of shift $\bar{p}$, is not likely to be much larger than the a priori probability of shift p (this may be interpreted as reflecting the imprecision in the magnitude estimate). Because of this, the step estimation and certainty assessment procedure is repeated on the next few runs (say 5) after an alarm occurs, to get a better estimate of the shift magnitude, and to allow the posterior probability to settle down. If a step really did occur, the posterior probability should settle down at a value of 1, but if the alarm was caused by random fluctuations, the posterior probability should settle down at a value of 0.

- Alarms are generated not only when the prediction error for the current run is outside the 3 σ limits, but also when one of the following *Western Electric rules* ([Wes65]) is satisfied:
  - Two of the last three points are between 2 and 3 σ from the mean, on the same side of the mean.
  - Four of the last five points are between 1 and 2 σ from the mean, on the same side of the mean.
  - The last 8 points are on the same side of the mean.

The complete rapid mode algorithm is as follows:

```
if one of the Western Electric rules signaled an alarm on the current
    run, or less than $h$ runs ago then
    if the last estimated shift occurred less than $l$ runs ago then
        remove the most recent adjustment to the intercept term.
    end if
    Estimate the shift magnitude $\Delta a$ and location $m$.
    Compute the posterior probability of shift $\bar{p}$.
    Adjust the intercept estimate by $\bar{p}\Delta a$.
    comment: Sometimes, e.g. in the case of a maintenance opera-
        tion, it may be known that a shift did occur. In this case, set
        $\bar{p}$ to 1, and $m$ to the number of runs since the shift occurred.
end if
```

The final section of this chapter will describe potential enhancements to the rapid mode algorithm.

5.3 Enhancements

The development of the algorithm just described employed two important simplifying assumptions, namely that control actions taken by the gradual mode algorithm can be ignored, and that the prior distribution of the shift magnitude has positive probability only for magnitudes of zero and Δa, the estimated magnitude. A possible way of accounting for the control actions taken in gradual mode has already been described. This section will describe how the second assumption might be relaxed.

Instead of assuming that the shift magnitude τ has a discrete prior probability distribution, one could in principle assume any kind of distribution. However, the application of Bayes' law can encounter numerical difficulties if the distribution is too complex. But if the prior distribution of τ is normal, and the z_i's are normally distributed, then the posterior distribution of τ will also be normal. This can be verified by substitution into Bayes law:

$$f_\tau(u|D_t) = \frac{\phi(z_{t-m+1},\ldots,z_t|\tau = u)f_\tau(u|D_{t-m})}{\int_{-\infty}^{\infty}\phi(z_{t-m+1},\ldots,z_t|\tau = u)f_\tau(u|D_{t-m})du}$$

where $\phi(\cdot)$ is a multivariate normal, with each component having mean τ and variance σ^2, and the components being independent. This is also a well known result in Bayesian statistics, see for example [WH89].

In particular, if the prior distribution of τ is normal with mean u_0 and variance s_0^2, then the posterior distribution will be normal with mean u_1 and variance s_1^2, where

$$\begin{aligned} u_1 &= \frac{\sigma^2}{ms_0^2+\sigma^2}u_0 + \frac{ms_0^2}{ms_0^2+\sigma^2}\frac{1}{n}\sum_{i=t-m+1}^{t} z_i \\ s_1^2 &= \frac{\sigma^2 s_0^2}{ms_0^2+\sigma^2} \end{aligned}$$

So the posterior mean u_1 is a weighted average of the prior mean u_0 and the average of $z_{t-m+1},\ldots,z_t$. It has been assumed here that an estimate m of the number of runs since a shift occurred is available.

To determine by how much the intercept term should be adjusted to minimize the expected mean squared deviation on the next run, we proceed as before: Let c be the amount by which the intercept term is adjusted. We subtract the equation used to determine the recipe $\mathbf{x}_{t+1}$

$$T = a + c + \mathbf{b}'\mathbf{x}_{t+1}$$

from the equation describing the behavior of the process

$$y_{t+1} = a + \tau + \mathbf{b}'\mathbf{x}_{t+1} + \epsilon_{t+1}$$

to get an expression for the deviation from target

$$y_t - T = \tau - c + \epsilon_{t+1}$$

It follows that

$$\begin{aligned} \mathrm{E}[(y_{t+1} - T)^2 | D_t] &= \mathrm{var}[y_{t+1} - T | D_t] + (\mathrm{E}[y_{t+1} - T | D_t])^2 \\ &= \mathrm{var}[\tau | D_t] + \sigma^2 + (\mathrm{E}[\tau | D_t] - c)^2 \end{aligned}$$

which shows that it is optimal to adjust the intercept term by the posterior expected shift magnitude $\mathrm{E}[\tau | D_t] = u_1$.

Further enhancements to the rapid mode algorithm are described in [SHI91]. They include using a bimodal prior distribution for the magnitude of shift, and taking advantage of the significant change in the process recipe which often accompanies rapid mode, to update the process sensitivity estimates in **b**.

Chapter 6

Stability and Robustness

A fundamental issue in designing any control system is the stability of the system. In this chapter, conditions are derived that guarantee that the output sequence $\{y_t\}$ of a process approaches its target value, when controlled by the RbR controller, under different assumptions about the process. By "approaching" will be meant that the sequence $\{\mathrm{E}[y_t]\}$ converges (preferably to T), and the sequence $\{\mathrm{var}[y_t]\}$ is bounded.

The analyses performed apply solely to the gradual mode of the algorithm. The results that follow may be interpreted as implying that if the algorithm is given an initial process model and recipe, and if the stated conditions are satisfied, then the sequence of recipes generated will approach a recipe for which the expected output value is the target value T. The initial model and recipe may be the result of rapid mode calculations, or they may be the initial information provided by the user.

The purpose of this analysis is to gain insight into how applicable the proposed algorithm is to different situations, and to see how sensitive it is to certain departures from the hypothesized process behavior. In particular, we are interested in sensitivity to the assumption of constant first-order coefficients, and in sensitivity to the noise and disturbance processes.

6.1 Motivation

As an example, let us investigate the stability of a deadbeat controller when applied to a hypothetical first order single input, single output process with no noise. A deadbeat controller is one where the control response is based only on the last output measurement, as opposed to a filtered version of the output sequence.

The process is shown graphically in figure 6-1. The shaded line represents the dependence of the output on the input. Also shown is the target value for the output and the recipe (input setting) at which the output would be on target. In figure 6-2, a change in the process is shown. The line representing the process has shifted but the process sensitivity (the slope of the line) has not changed. Also shown is the first output measurement obtained after the shift takes place. Figure 6-3 shows how the deadbeat control action is determined: the process recipe is changed based on the last output measurement and on an estimate of the process sensitivity. Here, the estimate used is 60% of the true process slope. The intersection of a line going through the last data point, with the estimated slope ("the process model") and a horizontal line through the target determines where the first recipe after the shift will be. But since the estimated slope is different from the true slope, the next output measurement will be below target, as shown by data point 1 in figure 6-3. In figure 6-4, the process model has been applied again to arrive at the second recipe after the shift. The corresponding output measurement is above target. As is clear from figure 6-5, the controller will cause the input to eventually converge on the proper recipe, where the output is on target.

In contrast, suppose that the estimated slope was 40% of the true slope. Figure 6-6 shows that output measurements 0,1, and 2 are further and further away from target, and the output sequence diverges. Figure 6-7 depicts a marginally stable process in which the controller cycles between two recipes, neither converging nor diverging. For the deadbeat controller, this occurs when the estimated slope is 50% of the true process slope. This example reveals that even without noise, the controlled process may be unstable. The first goal of this section is to analyze by how much the estimated process sensitivities can differ from their true values before the process becomes unstable.

Given that a stable controller has been designed, the next concern is how quickly the process converges. The second goal of this section is to determine how the speed of convergence affects the variability of the process output.

This analysis will apply only to the gradual mode of the algorithm, and the possibility of process shifts will be ignored. It will be assumed that the EWMA method

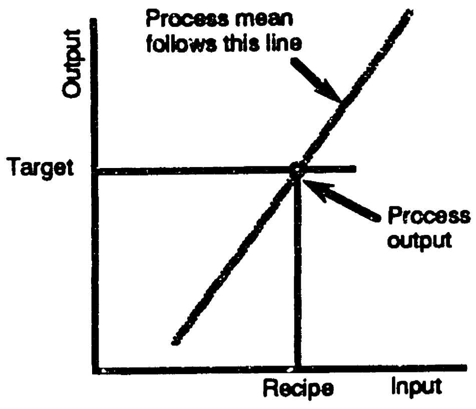


Figure 6-1: The shaded line represents a single input, single output process. The horizontal line goes through the target value for the output and the vertical line shows the recipe that would cause the output to be on target.

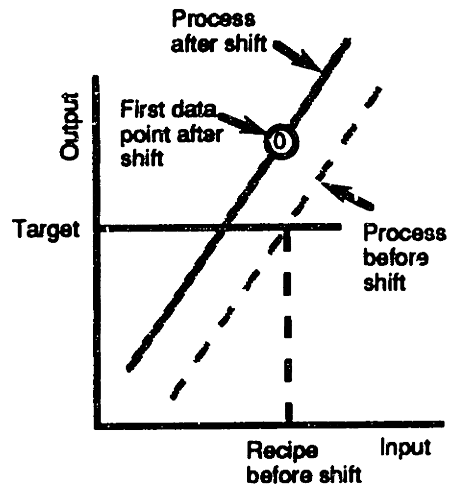


Figure 6-2: The line representing the process has shifted vertically, but the slope has not changed. The point labeled 0 is the first output measurement after the shift.

is used to update the process level. The analysis will proceed as follows: A set of difference equations describing the evolution of the controlled process will be set up for three kinds of postulated process behavior. These equations will then either be solved explicitly or otherwise analyzed to determine conditions that guarantee that the process will be stable in the sense described above. The set of difference equations will consist of the updating equation for the intercept term,

$$a_t = w(y_t - \mathbf{b}'\mathbf{x}_t) + (1 - w)a_{t-1} \tag{6.1}$$

the equation used to select a new recipe.

$$\mathbf{x}_t = \frac{T - a_{t-1}}{||\mathbf{b}||^2}\mathbf{b} + \left(\mathbf{I} - \frac{\mathbf{b}\mathbf{b}'}{||\mathbf{b}||^2}\right)\mathbf{x}_{t-1} \tag{6.2}$$

and a relation describing the postulated process behavior.

First, the case of a deterministic process will be analyzed. While this would usually be an inadequate description of a real manufacturing process, the results are easier to

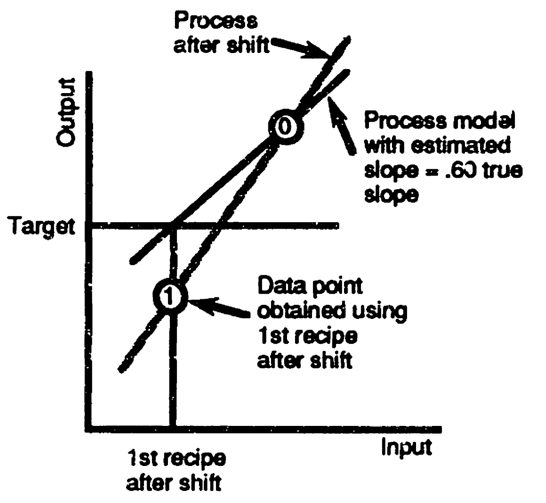

Process after shift
Process model with estimated slope = .60 true slope
Output
Target
Data point obtained using 1st recipe after shift
1st recipe after shift
Input

Figure 6-3: The solid line which goes through the point labeled 0 is the process model used by the deadbeat controller. Its slope is 60% of the true process slope. The vertical line represents the first recipe after the shift and the point labeled 1 is the resulting output measurement.

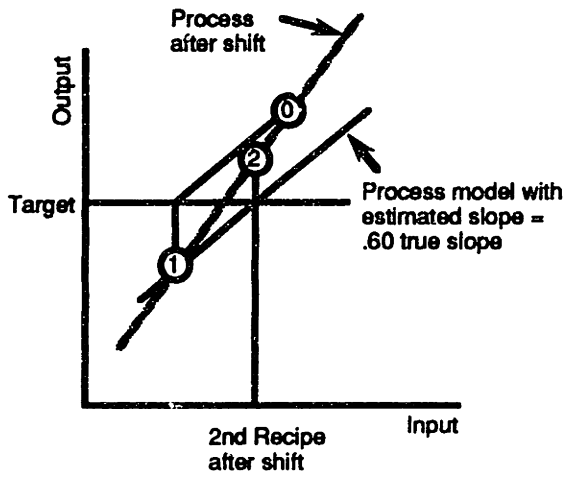


Figure 6-4: Here the deadbeat controller has been applied again to get the second recipe after the shift and the corresponding output measurement.

interpret, and do generalize to some extent to the stochastic process considered later.

6.2 First Order Deterministic Process

Suppose that the process behaves according to

$$y_t = \alpha + \beta' \mathrm{x}_t \tag{6.3}$$

where α and β are constants. This equation, along with the updating equations 6.1 and 6.2, completely specifies the evolution of the controlled process. Since x_t is a solution to the equation $T = a_{t-1} + \mathrm{b}'\mathrm{x}_t$, the updating equation 6.1 can be rewritten as

$$a_t = w(y_t - \mathrm{b}'\mathrm{x}_t) + (1 - w)a_{t-1}$$

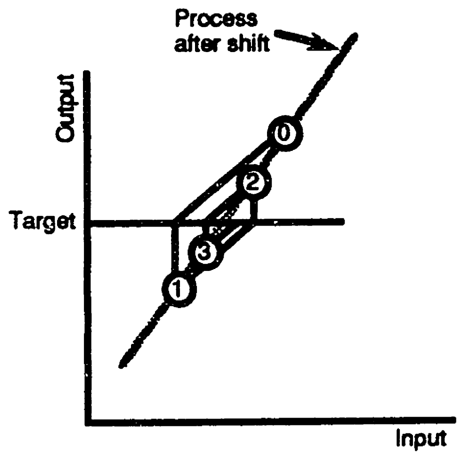


Figure 6-5: This figure illustrates how the process output will converge on target when the deadbeat controller uses a slope estimate that is 60% of the true slope.

$$= w(y_t - T + a_{t-1}) + (1 - w)a_{t-1}$$

Using 6.3, this can be written as

$$a_t - a_{t-1} = w(\alpha - T + \beta'\mathbf{x}_t) \tag{6.4}$$

Next, pre-multiply equation 6.2 with β', to get

$$\begin{aligned}
\beta'\mathbf{x}_t &= (T - a_{t-1})\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}} + \beta'\mathbf{x}_{t-1} - \frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}}\mathbf{b}'\mathbf{x}_{t-1} \\
&= (T - a_{t-1})\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}} + \beta'\mathbf{x}_{t-1} - (T - a_{t-2})\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}} \\
&= \beta'\mathbf{x}_{t-1} - \frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}}(a_{t-1} - a_{t-2})
\end{aligned}$$

By using 6.4, with t replaced by $t-1$, this reduces to

$$\begin{aligned}
\beta'\mathbf{x}_t &= \beta'\mathbf{x}_{t-1} - w\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}}(\alpha - T + \beta'\mathbf{x}_{t-1}) \\
&= \left(1 - w\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}}\right)\beta'\mathbf{x}_{t-1} - w\frac{\beta'\mathbf{b}}{\mathbf{b}'\mathbf{b}}(\alpha - T)
\end{aligned}$$

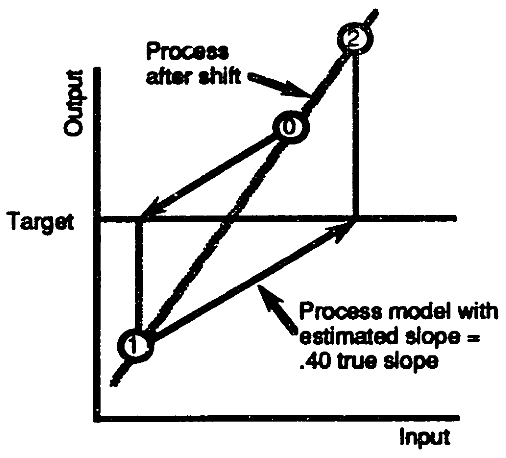


Figure 6-6: Here, a slope estimate that is 40% of the true slope is used. The output of the process when controlled by the deadbeat controller diverges in this case.

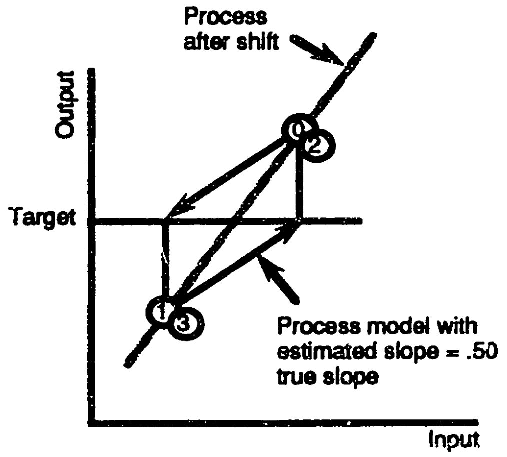


Figure 6-7: When the deadbeat controller uses a slope estimate that is 50% of the true slope, then the process output will be marginally stable, cycling between two values as shown.

The solution to this difference equation, (which defines the behavior of the sequence $\{\beta' \mathrm{x}_t\}$) is (see [Lue79])

$$\begin{aligned}
\beta' \mathrm{x}_t &= \theta^{t-1}\beta'\mathrm{x}_1 - (1-\theta)(\alpha - T)\sum_{i=0}^{t-1}\theta^{t-i} \\
&= \theta^{t-1}\beta'\mathrm{x}_1 - (\alpha - T)(1-\theta^t) \\
&= T - \alpha + \theta^{t-1}(\beta'\mathrm{x}_1 - \theta(T-\alpha))
\end{aligned}$$

where $\theta = 1 - w(\beta'\mathrm{b})/(\mathrm{b}'\mathrm{b})$. It follows that $\alpha + \beta'\mathrm{x}_t$ converges to T if and only if the ratio of convergence θ has absolute value less than one, which is equivalent to the condition

$$0 < w\frac{\beta'\mathrm{b}}{\mathrm{b}'\mathrm{b}} < 2$$

In the single input case, this condition reduces to

$$0 < w\beta/b < 2$$

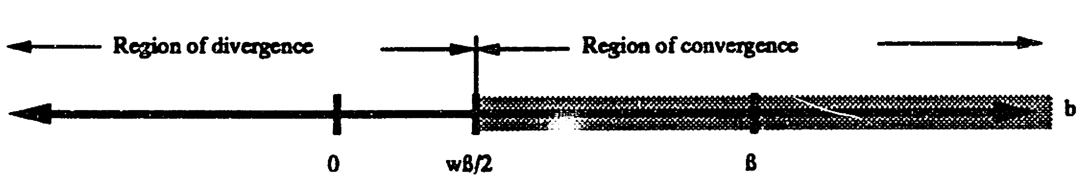


Figure 6-8: The region of convergence for the estimated slope b is the shaded interval $(w\beta/2, \infty)$. In this figure, $w = 2/3$ and β is positive.

This condition implies that if b is of the same sign and larger (in absolute value) than β, then stability is guaranteed. If b is of the same sign and smaller (in absolute value) than β, it must be larger (in absolute value) than $w\beta/2$ for stability. If b and β have different signs, the process will always be unstable. Figure 6-8 illustrates the region of convergence for b. When the stability condition holds, $|y_t - T|$ converges to zero as a geometric series, with convergence ratio θ. Notice that as w approaches 0, the convergence ratio θ approaches 1, implying that the speed of convergence decreases with w. On the other hand, it is evident from figure 6-8 that as w decreases, the range $(w\beta/2, \infty)$ in which b can lie expands.

Example

The following regression equations relating the predicted difference between left and right thickness measurements (y_t) to the flow balance (x_t) were obtained from data before and after a preventive maintenance operation was performed on an epitaxy reactor (see [SHIL91] and [IS91] for more details on this example and the two input example which follows):

$$\begin{aligned}\hat{y}_t &= 0.051 + 0.056x_t \text{ before maintenance} \\ \hat{y}_t &= -0.01 + 0.155x_t \text{ after maintenance}\end{aligned}$$

Suppose that a value of 0.056 was used for b, and that the regression equations captured the true behavior of the process. Then the convergence ratio would be $|1 - w|$ before and $|1 - 2.8w|$ after maintenance. So for $w > 0.72$, the y_t sequence,

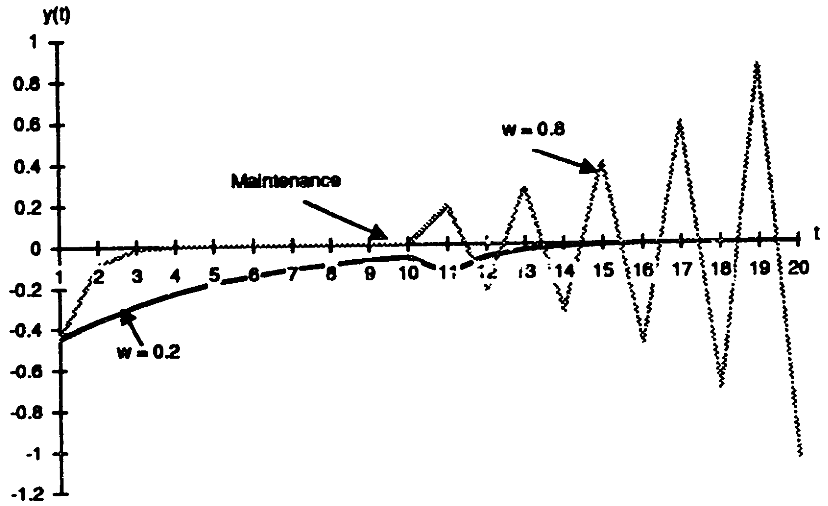


Figure 6-9: A simulation of the left-right thickness difference for 10 runs before and 10 runs after a maintenance operation was performed on an epitaxy reactor. When a value of 0.2 is used for w, the sequence converges slowly both before and after the maintenance. But when $w = 0.8$, the sequence converges quickly before and diverges after the maintenance.

after maintenance, would diverge. In figure 6-9, two y_t sequences are simulated for 10 runs before and 10 runs after the maintenance, using values of 0.2 and 0.8 for w. □

In the multiple input case, the convergence conditions can be rewritten as

$$\beta'\mathbf{b} > 0 \text{ and } ||\mathbf{b} - (w/4)\beta|| > (w/4)||\beta||$$

The first inequality implies that the angle between β and **b** must be less than 90 degrees. The second inequality guarantees that "**b** is not too small relative to β". In particular, **b** must be outside a sphere with radius $(w/4)||\beta||$, centered at $(w/4)\beta$.

Example

When the left to right thickness difference in the epitaxial growth process experiment was regressed on both the flow balance ($x_t(1)$) and the jet angle ($x_t(2)$), the following prediction equations weie obtained, before and after a maintenance operation:

$$\begin{aligned} y_t &= 0.051 + 0.056x_t(1) - 0.019x_t(2) \text{ before maintenance} \\ y_t &= -0.01 + 0.155x_t(1) + 0.014x_t(2) \text{ after maintenance} \end{aligned}$$

If these equations represented the true process behavior, then β'_{after} would equal [0.155, 0.014]. For the case when $w = 1$, the stability conditions reduce to

$$0 < 155b_1 + 14b_2 \text{ and } (b_1 - 0.0388)^2 + (b_2 + 0.0035)^2 > 0.0015$$

The convergence region is shown in figure 6-10. Also shown in the figure is the vector β_{before}. If this vector were used as an estimate of β_{after}, then the process would be unstable, for $w = 1$. The convergence region in figure 6-10 consists of three shaded regions: one in which the signs of both components of the slope vector β and the estimate **b** are the same, one in which the signs of β_1 and b_1 a.e opposite, and one in which the signs of β_2 and b_2 are opposite. This stands in contrast with the single input case examined before, where the signs of the estimated slope b and the true slope b had to be the same for stability to be guaranteed. However, in the multiple input case, the vector of estimates **b** can have some components with signs opposite to the respective components of the true slope vector b. For instance, this will occur whenever one component of **b**, say b_i, is significantly smaller (in absolute value) than the other components, since the contribution $b_i\beta_i$ to the inner product $\mathbf{b}'\beta$ of the small component will not be significant unless b_i is large. In practical terms, this means that if several input variables are being used to control a process, and one of those variables has almost no effect on the output which is to be controlled, then the estimated process sensitivity with respect to that input variable has little influence on the stability of the process output. □

6.3 Deterministic Second Order Process

For a real process, the relation between input and output is unlikely to be perfectly linear. To explore the effect of more complicated process behavior, suppose that the

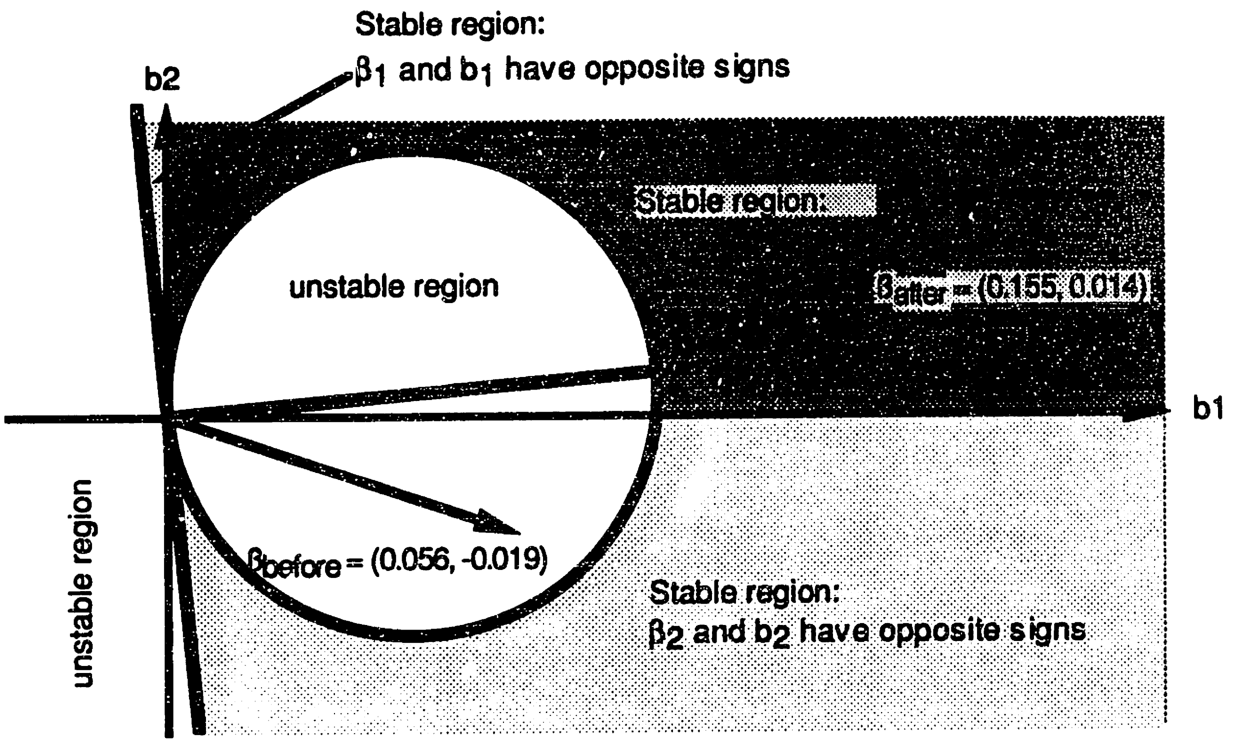


Figure 6-10: The region of convergence for [b1,b2] when b' = [0.155, 0.014] is the shaded area on the figure.

process is quadratic and has a single input, i.e.

$$y_t = \alpha + \beta x_t + \delta x_t^2 \tag{6.5}$$

Here, the evolution of a_t is governed by a difference equation of the form $a_t = Aa_{t-1}^2 + Ba_{t-1} + C$, where A, B, and C are constants. Depending on the values of the constants, the sequence $\{a_t\}$ might converge, diverge, or exhibit chaotic behavior.

One might expect that in order for $\{a_t\}$ to converge, b would have to be close to the first derivative of y_t, as defined by 6.5, evaluated at a point where $y_t = T$. Call the absolute value of this derivative μ. The value of μ^2 can be found by substituting the solutions to $\alpha + \beta x_t + \delta x_t^2 = T$ into $\mu = \frac{d}{dx_t} y_t \Big|_{y_t = T} = \beta + 2\delta x_t$. the result is $\mu^2 = \beta^2 - 4\delta(\alpha - T)$.

By substituting 6.5 and the recipe selection equation $x_t = (T - a_{t-1})/b$ in the intercept term updating equation $a_t = w(y_t - bx_t) + (1 - w)a_{t-1}$, it can be shown that the constants A, B, and C have the values

$$\begin{aligned} A &= w\delta/b^2 \\ B &= 1 - w\beta/b - 2w\delta T/b^2 \\ C &= w(\alpha + T\beta/b - T + T^2\delta/b^2) \end{aligned}$$

The analysis relies on the fact that any two quadratic functions $P(a)$ and $Q(a)$ are conjugate to each other in the sense that there exists an invertible mapping L, such that for any $a \in \Re$, the identity $L(Q(a)) = P(L(a))$ will hold.

In particular, suppose that the two quadratic functions are $P(a) = Aa^2 + Ba + C$, $Q(a) = a^2 + D$ and the mapping $L(a)$ is linear. Equating coefficients in the identity $L(Q(a)) = P(L(a))$ results in a formula for the mapping L:

$$L(a) = \frac{a - B/2}{A}$$

and the useful identity $1 - 4D = (B-1)^2 - 4AC$.

In [Dev90], the quadratic function $Q(a)$ is analyzed and it is shown that $Q^n(a_0)$ converges if $-3/4 < D < 1/4$ and a_0 is in the basin of convergence $(-a', a')$, where $a' = 1/2 + \sqrt{1-4D}/2$. The corresponding conditions for $P(a)$ are:

$$\begin{aligned} & -3/4 < D < 1/4 \\ \Leftrightarrow\ & 0 < 1 - 4D < 4 \\ \Leftrightarrow\ & 0 < (B-1)^2 - 4AC < 4 \end{aligned}$$

and

$$L(-a') < a_0 < L(a')$$

If the values of the constants A, B, and C are substituted, and the identity $\mu^2 = \beta^2 - 4\delta(\alpha - T)$ is used, the convergence conditions reduce as follows:

$$\begin{aligned} & 0 < (B-1)^2 - 4AC < 4 \\ \Leftrightarrow\ & 0 < (-2\beta/b - 2w\delta T/b^2)^2 - \\ & (4w\delta/b^2)(w(\alpha + T\beta/b - T + T^2\delta/b^2)) < 4 \end{aligned}$$

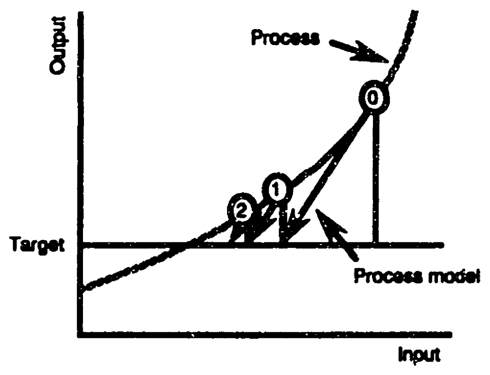


Figure 6-11: This figure illustrates how the output from a quadratic process converges to target when controlled by a deadbeat controller. Here, the estimated slope is larger than the true slope is in the region around the target.

$$\begin{aligned}
&\Leftrightarrow \quad 0 < (w/b)^2(\beta^2 - 4\delta(\alpha - T)) < 4 \\
&\Leftrightarrow \quad 0 < (w\mu/b)^2 < 4 \\
&\Rightarrow \quad 0 < \left|\frac{w\mu}{b}\right| < 2
\end{aligned}$$

and the starting value a_0 must be between

$$\frac{w}{2b\delta}(\mu + \beta + 2\delta T/\beta) \text{ and } -\frac{w}{2b\delta}(\mu + \beta + 2\delta T/\beta)$$

Note that the this convergence condition is similar to the single input deterministic linear process convergence condition. Also note, that as the quadratic coefficient δ approaches zero, the basin of convergence for a_0 approaches the interval $(-\infty, \infty)$.

Figures 6-11, 6-12, and 6-13 illustrate how the behavior of a process controlled with a deadbeat controller becomes more complicated when the process has some curvature. In figure 6-11, where the estimated slope is larger than the process slope in the region of operation, the output sequence converges much as it would if the process were linear. But when the estimated slope is smaller than the process slope is in the region around the target, as in figure 6-12, the output sequence may go through a cycle of values. Depending on the process parameters, this limit cycle can be arbitrarily long. This type of process behavior may be thought of as a gray area between convergence and divergence. If the estimated slope is too small the output sequence will diverge, as in figure 6-13.

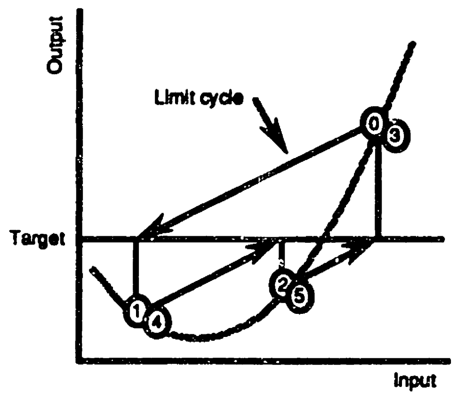


Figure 6-12: When the process is non-linear, the behavior of the output sequence can become arbitrarily complex. This figure illustrates how a limit cycle can result when the estimated slope is smaller than the true process slope is in the region around target.

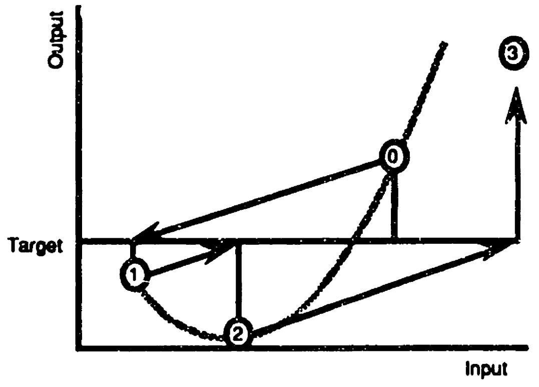


Figure 6-13: This figure illustrates a divergent output sequence of a quadratic process, controlled by a deadbeat controller using an estimated process slope that is too small.

One might argue that the situation depicted in figure 6-12, where the sign of the process slope changes in the region of operation is unlikely to occur for a manufacturing process. More research is needed to determine the implications for process stability of moderate departures from linearity in the region of operation.

6.4 Drifting First Order Process

The simple deterministic process models analyzed in the last two subsections probably do not adequately describe any real manufacturing process, which will be subject to various disturbances, some of which are unpredictable. A more reasonable model would include a random error term ϵ_t, to represent measurement error and other unpredictable sources of noise, and a term representing drift in the intercept term. Suppose the intercept term changes by an amount δ_t between successive runs, where $\{\delta_t\}$ is an iid sequence with $E[\delta_t] = d\sigma$ and $\text{var}[\delta_t] = (r\sigma)^2$. In other words, the process behaves according to

$$y_t = \alpha_0 + \sum_{i=1}^{t} \delta_i + \beta x_t + \epsilon_t \tag{6.6}$$

Note that a process subject to purely deterministic drift is obtained as a special case (by setting $r = 0$) and a purely random drift is obtained by setting $d = 0$.

As in the last subsection, only single input processes will be considered here. If y_t, as defined above in 6.6, and $x_t = (T - a_{t-1})/b$ are substituted in the intercept updating equation $a_t = w(y_t - bx_t) + (1 - w)a_{t-1}$, the difference equation

$$a_t = \theta a_{t-1} + \gamma_0 + w\epsilon_t + w\sum_{i=1}^{t} \delta_i$$

is obtained, where θ equals $1 - w\beta/b$ as before, and $\gamma_0 = w(\alpha_0 + t\beta/b - T)$. The solution to this difference equation ([Lue79]) is

$$a_t = \theta^t a_0 + \frac{1 - \theta^{t+1}}{1 - \theta}\gamma_0 + w\sum_{i=1}^{t} \theta^{t-1-i}\epsilon_i + w\sum_{i=1}^{t} \theta^{t-i}\sum_{j=1}^{i} \delta_j$$

Substituting this solution in 6.6 and using $x_t = (T - a_{t-1})/b$, one obtains

$$y_t = T + \theta^t\left(\alpha_0 - T + \frac{T\beta}{b} + \frac{a_0\beta}{b\theta}\right) + \sum_{i=1}^{t} \theta^{t-i}\delta_i - (1 - \theta)\sum_{i=1}^{t-1} \theta^{t-1-i}\epsilon_i + \epsilon_t$$

Before analyzing this expression further, consider each of the five terms on the right hand side. The parenthesized term represents the effect of of the initial estimate a_0, which dies out as $t \to \infty$, assuming that the convergence condition $|\theta| < 1$ holds. The first summation shows how the effect of any particular drift term δ_i will eventually die out, and the second summation shows how any particular noise term ϵ_i dies out. But since a new noise term ϵ_t is generated at each run, the sequence $\{y_t\}$ does not converge.

However, since $\mathrm{E}[\epsilon_i] = 0$, the sequence $\{\mathrm{E}[y_t]\}$ does converge, because

$$\mathrm{E}[y_t] = T + \frac{d\sigma}{1-\theta} + \theta^t \left(\alpha_0 - T + \frac{T\beta}{b} + \frac{a_0\beta}{b\theta} - d\sigma\right)$$

Assuming that the convergence condition is satisfied, the expected output will converge to $T + d\sigma/(1-\theta)$. The offset term $d\sigma/(1-\theta)$ reflects the fact that if the drift has a deterministic component ($d \neq 0$), then the controller will always be one step behind the process, in the sense that the controller reacts to the drift after it has occurred, rather than anticipating it (to do so would be feed-forward control).

The variance of y_t is

$$\mathrm{var}[y_t] = \sigma^2 \left\{ 1 + r^2 \frac{1-\theta^{2t}}{1-\theta^2} + (1-\theta)^2 \frac{1-\theta^{2t-2}}{1-\theta^2} \right\}$$

which converges to

$$\mathrm{var}[y_\infty] = \sigma^2 \left\{ 1 + \frac{r^2}{1-\theta^2} + \frac{1-\theta}{1+\theta} \right\}$$

Of particular interest is the asymptotic mean squared deviation from target $\mathrm{E}[(y_\infty - T)^2]$, which equals the sum of the asymptotic variance and the offset term $d\sigma/(1-\theta)$ squared. This quantity will be called msd_∞, and can be written as a function of the precision of the sensitivity estimate, as measured by b/β, the EWMA weight w, and the parameters r and d of the drift δ_i between successive runs:

$$\mathrm{msd}_\infty = \mathrm{var}[y_\infty] + (\mathrm{E}[y_\infty - T])^2 = \sigma^2 \left\{ \frac{2b/\beta}{2b/\beta - w} + \left(\frac{db/\beta}{w}\right)^2 + \frac{(rb/\beta)^2}{w(2b/\beta - w)} \right\}$$

The msd_∞ is minimized at σ^2 when there is no drift ($d = r = 0$), no control action is taken ($w = 0$), and the process is initially on target ($\mathrm{E}[y_1] = T$). We will now consider the inflation of msd_∞ over this lower bound for various special cases.

6.4.1 Deterministic Drift

If the drift is purely deterministic ($r = 0$), the asymptotic mean squared deviation becomes

$$\mathrm{msd}_\infty = \sigma^2 \left\{ \frac{2b/\beta}{2b/\beta - w} + \left(\frac{db/\beta}{w}\right)^2 \right\} \tag{6.7}$$

In figure 6-14, msd_∞ is shown as a function of w, for two different values for the drift d, and three different values of b/β. The variance of noise σ was set equal to 1. Each of the points shown on the figure represents the simulation of a controlled process with drift, for 100 runs. These simulations were performed to verify the formula for msd_∞, and the results agree closely. The figure suggest that if the process drift is less than 10% of the standard deviation of noise per run (i.e. $|d| < 0.1$) , and the slope estimate is thought to differ by no more than 50% from the true slope, then a value between 0.2 and 0.3 is a reasonable choice for w.

Figure 6-14 also illustrates some of the qualitative aspects of the asymptotic mean squared deviation: For any given non-zero values of d and b/β, the msd_∞ has a unique minimizing value of w. As d or b/β increases, the minimizing value of w increases, and as d increases, the minimum msd_∞ increases. In operational terms this means that the faster the process drifts, the more aggressively should the process be controlled and the larger will be the variability of the process output.

It is interesting to consider some limiting cases of msd_∞. When $d = 0$, and $b = \beta$, msd_∞ reduces to $2\sigma^2/(2 - w)$, which is minimized at $w = 0$. In words, if the process does not drift and the process sensitivity is known, then it is optimal to leave the process alone. However, note that this violates the convergence condition, so if the process is off target to begin with, it will remain off target. If w is set equal to 1, the msd_∞ is inflated by a factor of two, as explained by [Dem86] in the context of the funnel experiment. On the other hand, if d does not equal zero, then msd_∞ tends to infinity as $w \to 0$. This occurs because if a drifting process is not controlled, it will drift further and further away from target.

Example

The RbR controller was applied to an epitaxial growth process ([SHI91]). Before doing so, a designed experiment was performed to determine the sensitivity of the

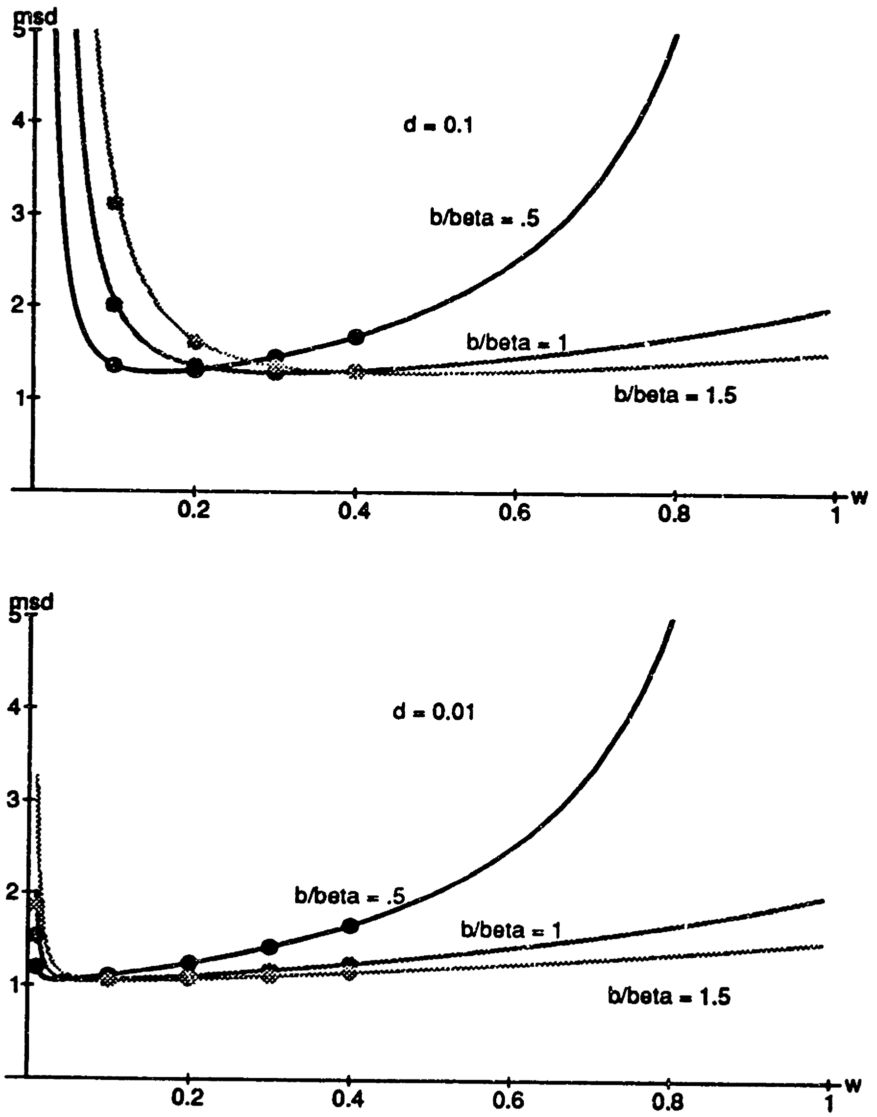


Figure 6-14: The mean squared deviation of the output from target has a unique minimum as a function of w. These two graphs illustrate that this minimum moves to the right as b/β increases. Also, the mean squared deviation is seen to grow faster when w is below the minimizing value than above it, suggesting that it is safer to use a w which is too large than too small. These curves were generated using $\sigma = 1$, $d = 0.1$ and 0.01; and $b/\beta = 0.5, 1.0$, and 1.5. The points in the figure represent the results of simulations which were performed to verify equation 6.7.

left to right thickness difference (y_t) to the balance of flow between the two nozzles (x_t). The following regression equation is based on data from that experiment:

$$y_t = -0.013 + 0.07x_t$$

From this equation, an estimate of 0.07 for the process sensitivity was obtained. This value was used for the algorithm parameter b, and a value of 0.1 was used for w. Figure 6-15 shows fifty consecutive output measurements of the left to right thickness difference in an epitaxial growth process controlled by the RbR controller. The root mean squared deviation from target (rmsd) for this sequence is 0.103.

To illustrate the dependence of the rmsd of the output sequence on the algorithm parameters b and w, three computer simulations were performed. These simulations were performed as if process behavior was characterized by $y_t = -0.013 + 0.07x_t + d\sigma t + \epsilon_t$, where the magnitude of drift d was set equal to 0.05 and the standard deviation of noise σ was set equal to 0.1, and then applying the RbR controller algorithm to this simulated process. Figure 6-16 shows a sequence where values of 0.1 for w and 0.07 for b (as for the real process) were used. The rmsd is 0.121, about 20% higher than for the real process, because of the superimposed drift. Equation 6.7 can be used to predict the inflation in rmsd and here the predicted increase is 14%. Figure 6-17 illustrates the case of under-control, meaning that the control action taken was not sufficient to keep up with the process drift. The simulation was performed using a low value of w (0.01), which causes the controller to adapt slowly to new measurements, and a process slope estimate that is twice as large as the true slope, causing the controller to underestimate the necessary control action. In this case, equation 6.7 predicts a 900% increase in rmsd, whereas an 83% increase is observed. The discrepancy occurs because equation 6.7 predicts the asymptotic inflation in rmsd, but the process shown in figure 6-17 has not yet reached steady state. In other words, the process will continue to drift away from target until the output is about 10 σ away from target. Figure 6-18 shows what happens when excessive control action is taken, i.e. over- control. To achieve this, the RbR controller used a high value of w (0.5), making the controller too sensitive to recent measurements, and a process slope estimate which is only 50% of the true slope, causing the controller to over-estimate the magnitude of the necessary control actions. The 38% increase in rmsd is in good

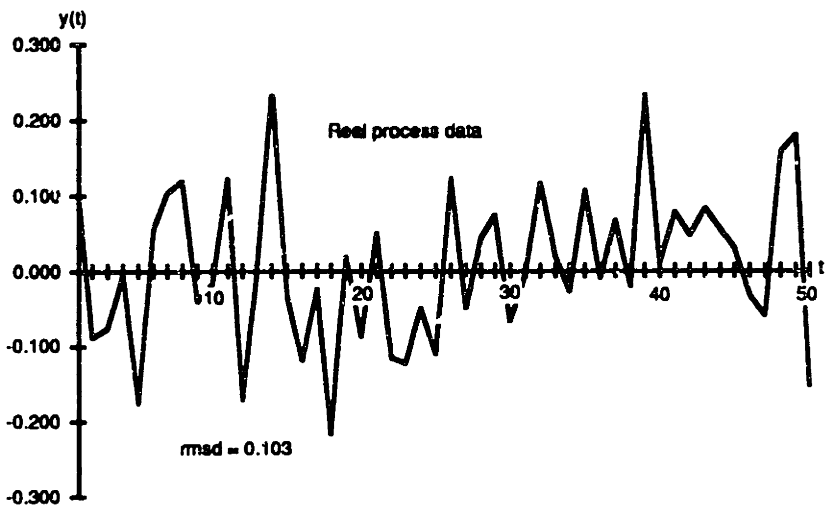


Figure 6-15: This plot shows fifty measurements of the left to right thickness difference for an epitaxial growth process controlled with the RbR controller.

agreement with the 42% predicted by equation 6.7. □

6.4.2 Random Drift

For purely random drift ($d = 0$), the asymptotic mean squared deviation becomes

$$\text{msd}_\infty = \sigma^2 \left\{ \frac{2b/\beta}{2b/\beta - w} + \frac{(rb/\beta)^2}{w(2b/\beta - w)} \right\}$$

In figure 6-19, msd_∞ is shown as a function of w, for two different values of r and three different values of b/β. For this magnitude of drift, it appears that a value between 0.1 and 0.2 would be a good choice for w. The comments made about the qualitative behavior of msd_∞ in the previous subsection apply to here also: As the ratio b/β, or the magnitude of drift r increase, the value of w which minimizes the msd_∞ increases, and the faster the process drifts, the larger will the minimum msd_∞ be. However, the msd_∞ increases less rapidly as w approaches zero than for the case of deterministic drift.

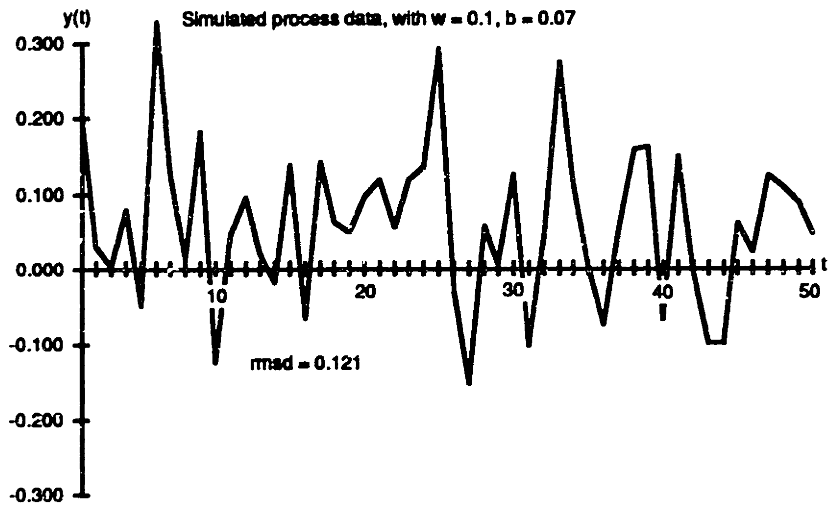


Figure 6-16: A simulated sequence of left to right thickness measurements, with a deterministic drift of 0.05 σ between successive runs. The same values were used for the RbR controller algorithm parameters as for the real process shown in figure 20. The root mean squared deviation from target is about 21% larger than for the real process, whereas equation 6.7 predicts a 14% increase.

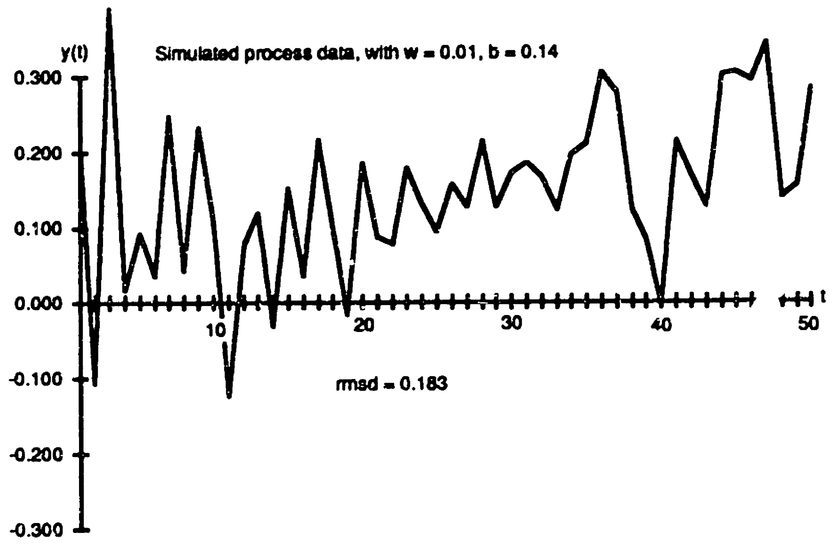


Figure 6-17: This process simulation illustrates the case of under-control. The value used for the estimated slope is 100% larger than the true slope and the EWMA weight is only 0.01. The output sequence is seen to be drifting away from target and has not reached a steady state after 50 runs. The root mean squared deviation is 83% larger than for the real process, which may be compared to a 900% increase in rmsd predicted by equation 6.7, suggesting that the output will continue to drift to about 10 σ away from target before settling down.

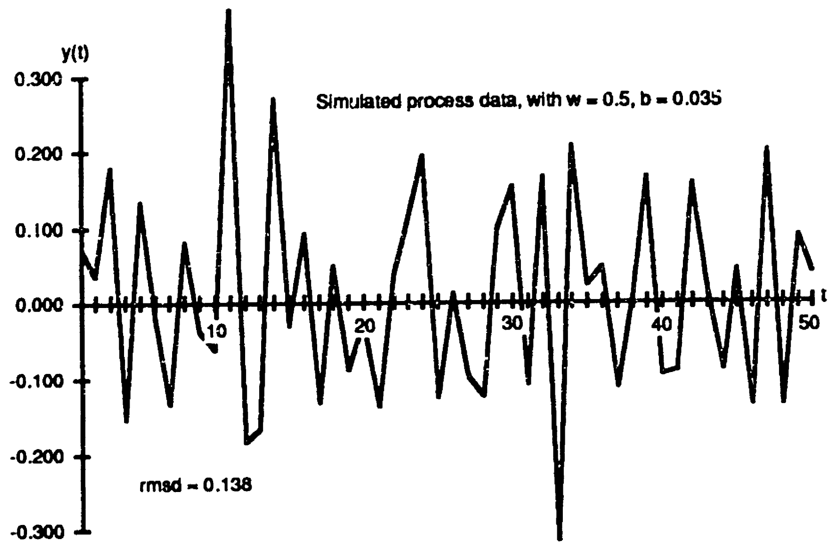


Figure 6-18: A simulated output sequence illustrating over-control. The process slope is underestimated by 50%, and an EWMA weight of 0.5 is used, which gives a weight of 50% to the last observed measurement. The process drift is kept in check, but the control action is excessive, resulting in a 38% increase in rmsd. Equation 6.7 predicts a 42% increase.

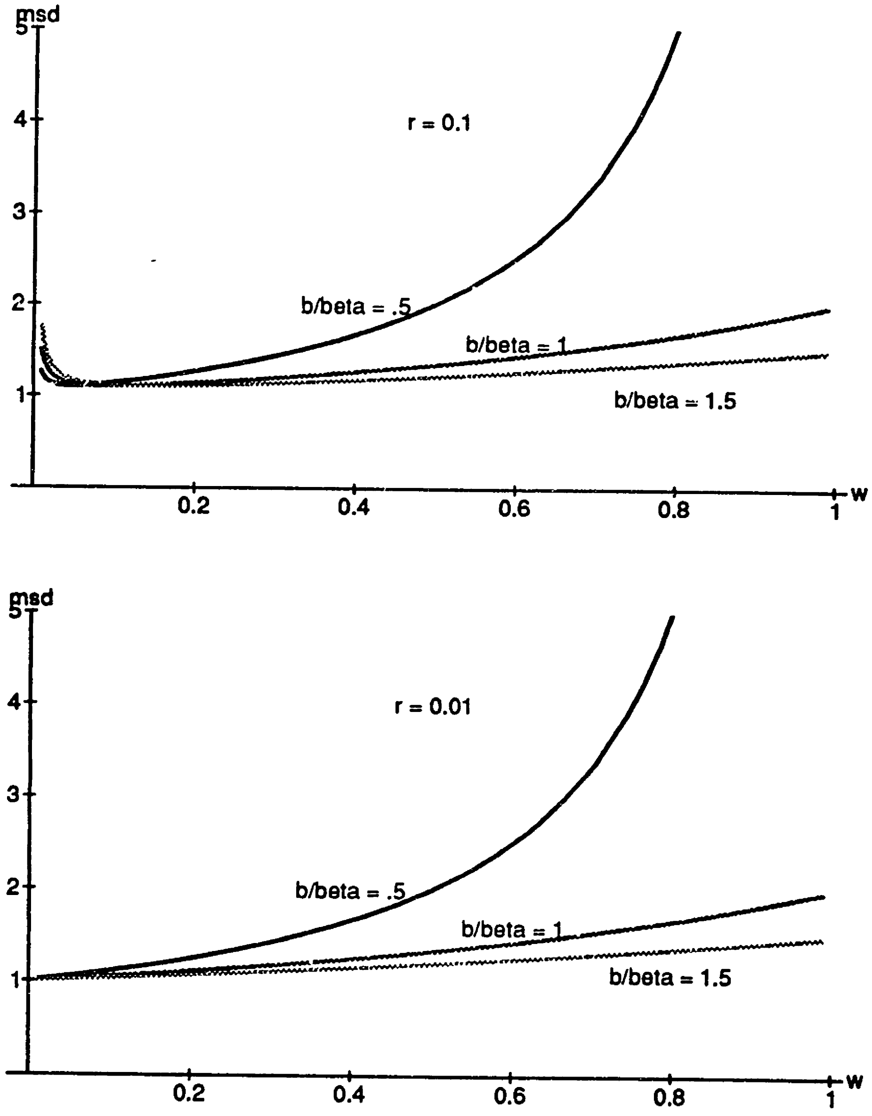


Figure 6-19: The asymptotic mean squared deviation, as a function of w, is seen to have a unique minimum. These curves were generated using $\sigma = 1$, $r = 0.1$ and 0.01; and $b/\beta = 0.5$, 1.0, and 1.5. For this magnitude of drift, the minimizing value of w does not seem to depend much on the value of b/β. As in the deterministic drift case, the variability is seen to increase faster for values of w below the minimizing value.

Chapter 7

Testing the Algorithm

The performance of the algorithm was analyzed in the previous chapter. However, the analysis assumed that the process was first order (except for section 6.3), and applied only to the gradual mode of the algorithm. This chapter will report on a simulation experiment that was designed to test the performance of the RbR controller algorithm with both gradual and rapid modes enabled in controlling a second order process, which may be subject to both shifts and drifts.

7.1 Implementation of the Algorithm

The RbR controller algorithm described in this thesis has been implemented using the MATLAB software package ([MLB90]). This implementation, which has a menu-based interface and displays process data graphically, is described in [SGH+91].

The algorithm was configured to automatically control a simulated process for 200 runs, under 256 different conditions, to be described shortly. The basis for the simulated process was an equipment simulator which simulates the processing of one batch of wafers in an LPCVD reactor, using a finite difference approximation to the process dynamics. This equipment simulator is described in [SPG91]. Refer to figure 1-5 for schematic representation of an LPCVD reactor. This figure shows three injectors through which gas flows into the reactor tube: the load, center, and source injectors. Only the total gas flow rate (x_t) was varied, and the output variable to be controlled was the average growth rate (y_t). The following values were used for the other process inputs:

- temperature 625° C.
- pressure 0.205 torr.
- center injector position 92.7 cm.
- source injector position 121 cm.
- load injector flow rate 26.6% of the total flow rate.
- center injector flow rate 36.7% of the total flow rate.
- source injector flow rate 36.7% of the total flow rate.

Note from figure 1-5 that the position of the load injector is fixed. By running the simulator using total flow rate values between 130 and 170 sccm, the following empirical model of the process was estimated using least squares regression:

$$y_t = 49.6331 + 3.7047u_t - 0.1578u_t^2$$

where $u_t = (x_t - 150)/20$ is a coded version of the total flow rate. Because the equipment simulator takes a long time to run, and since the empirical model provided an almost perfect fit to the data, it was decided to use the above empirical relation to simulate the process to be controlled, rather than run the equipment simulator each time. Some limited experimentation suggested that the resulting process output, when controlled with the RbR controller, was indistinguishable from running the equipment simulator itself.

7.2 Experimental Design

An orthogonal experimental design with an inner and outer array was used to select the conditions under which the algorithm was to be run. The outer array contained factors which define the process behavior, and the inner array factors were parameters of the RbR controller algorithm to be tuned. The outer and inner array factors will now be described.

7.2.1 Outer Array

The following four factors were chosen as outer array factors:

1. Magnitude of deterministic drift, d, with levels 0 (no drift), 1 σ per 1000 runs, 1 σ per 500 runs, and 1 σ per 200 runs.
2. Magnitude of shift, with levels 0 (no shift), 1 σ, 2 σ, and 3 σ. The shift occurred between runs 100 and 101.
3. Process curvature, with four levels, which are specified below.
4. The difference between the true first order coefficient β and the estimated value b, with levels 0%, 10%, 30%, and 50%. These percentages are the amounts by which the estimated slope exceeds the true slope, i.e. $b = (1 + d/100)\beta$, where d is the percentage difference. It was shown in chapter 6 that overestimating the slope never causes instability, so it might have been more interesting to use underestimates of the slope. However, the simulation experiment was performed before the stability analysis was performed, so this was not known at the time.

The first two of these factors specify the disturbances to the process. The factor levels were chosen to represent values which might be encountered in practice. For both of those factors, one would expect that the larger the drift or shift, the worse should the process performance be.

The other two factors define by how much the simulated process differs from the idealized linear model used by the algorithm. One would expect that the larger the process curvature and the larger the difference between the estimated and true slopes, the worse should the controlled process perform.

The levels of the process curvature factor will now be specified. The measure of process curvature used is "the percentage difference between the process slopes at the two extremes of the operating space". Taking the derivative of the empirical process model described before, we get

$$y_t'(u_t) = 3.7047 - 0.3156u_t = 3.7047 - au_t$$

where a is parameter to be varied, to simulate different amounts of process curvature. If the slope of y_t at $u_t = -\Delta u$ and $u_t = 0$ differs by d percent, the following condition has to hold:

$$\begin{aligned}
\frac{d}{100} &= \frac{y_t'(-\Delta u) - y_t'(0)}{y_t'(0)} \\
&= \frac{3.7047 - a(-\Delta u) - 3.7047}{3.7047} \\
\Rightarrow a &= 3.7047 \times 10^{-2}\frac{d}{\Delta u}
\end{aligned}$$

The desired factor levels were $d = 0\%$, 10%, 30%, and 50%, and the value of Δu used was 1 (corresponding to 20 sccm). In the process simulations, the flow rate varied from 150 sccm to 115 sccm, for a step of size 3 σ. Thus the operating space had a width of 35 sccm, so a Δu corresponding to 20 sccm would cause the factor levels to be underestimated in this case. But when the process is simulated without any step the operating space would be narrower, and the factor levels would be overestimated. Hence the percentage values given should not be interpreted literally, but as approximate values.

The processes corresponding to the four levels of curvature are (in terms of the coded variable u_t):

$$\begin{aligned}
y_t &= 49.6331 + 3.7047u_t \\
y_t &= 49.6331 + 3.7047u_t - 0.1852u_t^2 \\
y_t &= 49.6331 + 3.7047u_t - 0.5557u_t^2 \\
y_t &= 49.6331 + 3.7047u_t - 0.9262u_t^2
\end{aligned}$$

In terms of x_t, the real flow rate, the processes are

$$\begin{aligned}
y_t &= 21.8479 + 0.1852x_t \\
y_t &= 11.4304 + 0.3241x_t - 0.000463x_t^2 \\
y_t &= -9.4103 + 0.6020x_t - 0.001389x_t^2 \\
y_t &= -30.2509 + 0.8799x_t - 0.002316x_t^2
\end{aligned}$$

Depending on the level of the process curvature factor, one of these four relations was used to simulate the process, with a normally distributed random noise term added to y_t at each run. The noise had mean zero and variance 1 ($\sigma = 1$).

For each combination of the outer array factors, four inner array factors, all parameters of the RbR controller algorithm, were varied. These factors will now be described.

7.2.2 Inner Array

The inner array factors are

1. The EWMA weight w, with levels 0.01, 0.1, 0.3333, and 0.5. The larger the value of this factor, the more responsive is the controller. Based on the analysis in chapter 6, one would expect that the best value of this factor would depend on the magnitude of drift, and perhaps the magnitude of shift.

2. How often rapid mode is engaged to re-estimate a shift after a generalized SPC alarm, with levels 1, 5, 10, and 20. The more often a shift is re-estimated, the more precise should the estimated shift parameters be, so for this factor, a larger value should be better.

3. The number of runs which have to elapse before a shift is "locked in", with levels 10, 20, 40, and 100. For this level one would also expect that larger values should be better. Note, however, that this depends on the assumption that shifts are infrequent, because if two shifts occur in a short interval of time, one would want to lock in the first shift before estimating the second one, rather than approximating both shifts by one.

4. The a priori probability of a shift occurring on any given run, with levels 0.0005, 0.005, 0.05, and 0.1. This factor determines how "conservative" the controller is in compensating for a step disturbance.

The first factor, w, was defined in chapter 4, and the other three were defined in chapter 5.

The first four columns of Taguchi's L_{16} orthogonal array ([Tag87]) were used both for the inner and outer arrays, for a total of 256 combinations of the outer and inner array factor levels. The first four columns of the L_{16} array are shown below (each column corresponds to a factor, and the numbers correspond to a factor level).

$$L_{16} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 3 & 3 & 3 \\ 1 & 4 & 4 & 4 \\ 2 & 1 & 2 & 3 \\ 2 & 2 & 1 & 4 \\ 2 & 3 & 4 & 1 \\ 2 & 4 & 3 & 2 \\ 3 & 1 & 3 & 4 \\ 3 & 2 & 4 & 3 \\ 3 & 3 & 1 & 2 \\ 3 & 4 & 2 & 1 \\ 4 & 1 & 4 & 2 \\ 4 & 2 & 3 & 1 \\ 4 & 3 & 2 & 4 \\ 4 & 4 & 1 & 3 \end{bmatrix}$$

For each experimental run, the root mean squared deviation from target (rmsd) and the average run length (arl) between generalized SPC alarms were computed. Those two summary statistics were used as measures of performance of the controlled process. In general, one would want the rmsd to be as small as possible, and the arl to be as long as possible (a short arl means that a large number of false alarms occurred).

7.3 Results

The results of the simulation experiment are shown in figures 7-1 to 7-4. Each figure shows the marginal effect of either the outer array or inner array factors on the rmsd or on the arl.

Figure 7-1 shows the marginal effects of the outer array factors on the rmsd. Each 'x' in the graphs in this figure (and the three figures that follow) represents the average rmsd over all inner array combinations, for a given outer array combination. The lines in the graphs connect the average rmsd's for a given level of a factor. For example, in the upper left graph in figure 7-1, the line connects the overall average rmsd's for a given level of drift.

Except for the magnitude of drift, which appears to have an insignificant effect on the rmsd, for each of the other factors (magnitude of shift, amount of process curvature, and difference between true and estimated slopes) large values of the factor tend to increase the rmsd, as expected. The following regression equation gives further support to this visual assessment:

$$\text{rmsd} = \overset{(0.02)}{1.08} - \overset{(3.28)}{2.68}\ \text{drift} + \overset{(0.01)}{0.02}\ \text{shift} + \overset{(0.0003)}{0.0009}\ \text{curvature} + \overset{(0.0003)}{0.03}\ \text{slope}$$

Here the regressor variables are the outer array factor levels, and the numbers in parenthesis are the standard errors associated with the regression coefficient below. The R^2 value was 88%.

In figure 7-2, the marginal effects of the outer array factors on the arl are shown. Large shift sizes are seen to be associated with shorter arl's, which is not surprising, since shifts cause alarms. Poor slope estimates are also seen to decrease the arl's. The effect of the drift magnitude and process curvature on arl appears to be insignificant. The regression analysis for this case resulted in

$$\text{arl} = \overset{(8.55)}{89.92} - \overset{(1766.12)}{841.46}\ \text{drift} - \overset{(2.96)}{10.60}\ \text{shift} - \overset{(0.17)}{0.09}\ \text{curvature} - \overset{(0.17)}{0.45}\ \text{slope}$$

with an R^2 of 59%.

Figure 7-3 show the marginal effects of the inner array factors on the rmsd. As expected, based on the analysis in chapter 6, the rmsd as a function of the EWMA weight w has a unique minimum, which appears to be close to 0.1. The effects of the number of re-estimations (re-est) and the number of runs before a shift is locked in on the rmsd appear insignificant. Note that if shifts were more frequent, then this might not be the case. The lowest value used for the a priori probability of shift (0.0005) causes the rmsd to increase, probably because the controller becomes too conservative in responding to the shift. The regression equation here was

$$\text{rmsd} = \overset{(0.01)}{1.18} + \overset{(0.02)}{0.07}\, w - \overset{(0.0006)}{0.0002}\ \text{re-est} - \overset{(0.0001)}{0.001}\ \text{lock-in} - \overset{(0.10)}{0.15}\ \Pr\{\text{shift}\}$$

and R^2 was 45%.

Finally, figure 7-4 shows the marginal effects of the inner array factors on the arl. Low values of the EWMA weight w are seen to increase the arl's, perhaps because under-control allows drift to build-up far enough to cause alarms. The number of re-estimations and the number of runs before a step is locked in have an insignificant effect, and so does the a priori probability of shift. The regression equation was

$$\text{arl} = \overset{(10.29)}{70.81} - \overset{(20.90)}{81.11}\, w + \overset{(0.57)}{0.07}\ \text{re-est} + \overset{(0.12)}{0.05}\ \text{lock-in} + \overset{(100.14)}{149.29}\ \Pr\{\text{shift}\}$$

and R^2 was 56%.

Summary of Results The results of the experiment are in agreement with the expectation that the more different the process is from the idealized linear process model, and the larger the disturbances to the process, the worse will the performance of the process be. However, the algorithm appears reasonably robust against those departures from the ideal process behavior, in the sense that none of the outer array factors caused the output variation, as measured by the rmsd, to change increase by more than 20%.

Also, the qualitative behavior of the rmsd as a function of the EWMA weight w appears to be the same as that derived analytically in chapter 6. This qualitative behavior is characterized by a unique minimizing value of w, with the rmsd growing quickly when w approaches 0, and growing slowly when w approaches 1.

The only unexpected feature of the results was the insignificant effect of the three rapid mode parameters, that were used as inner array factors (the number of re-estimations, number of runs before a shift is locked in, and the prior probability of shift), on the process performance. However, this can be explained by the fact that any improvement in the way the controller responds to a shift disturbance will be averaged over all 200 runs, since at most one shift occurred during any experimental run. Therefore, to assess the effects of these three factors more accurately, one might design an experiment with a higher frequency of shifts.

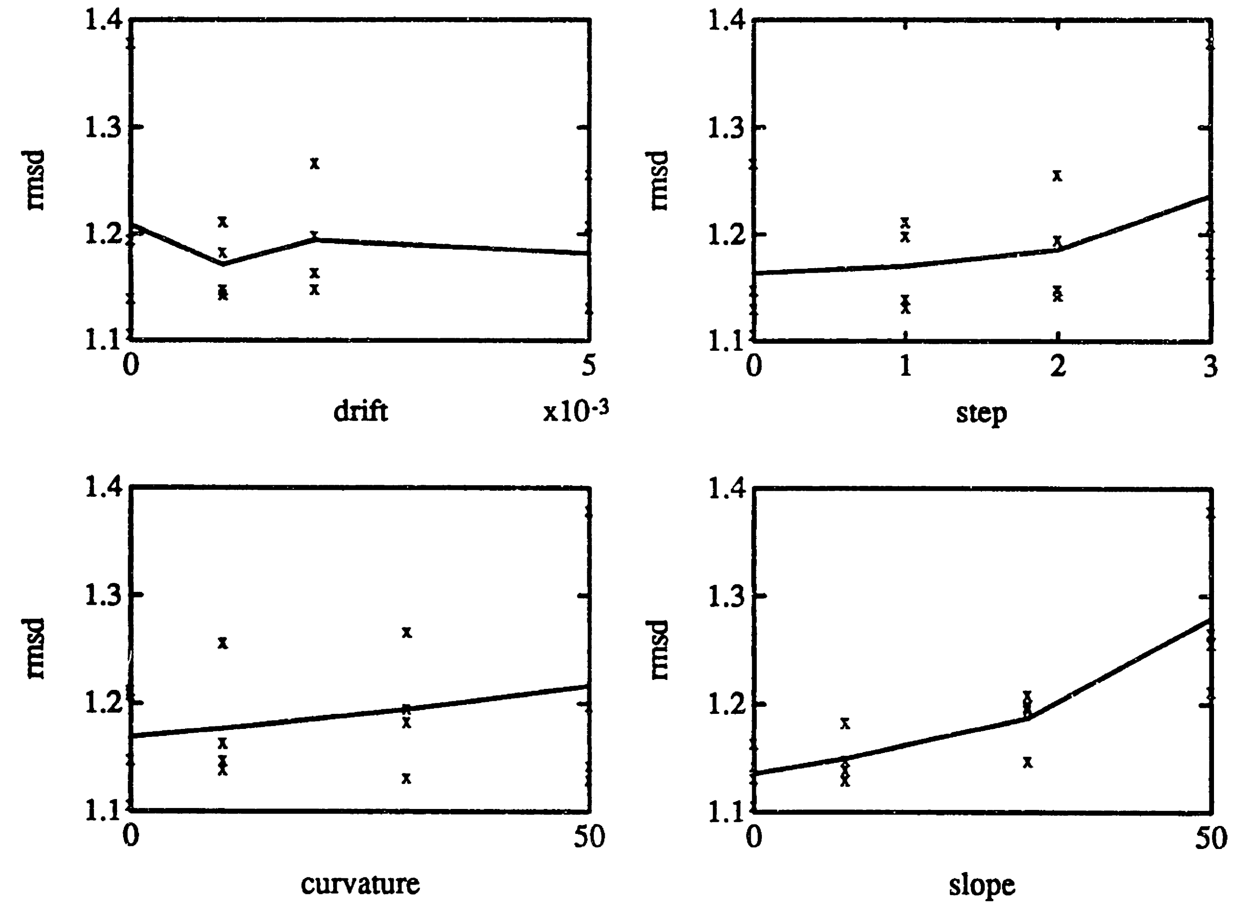


Figure 7-1: The marginal effects of the outer array factors on the root mean squared deviation from target, rmsd. In this, and the next three figures, each of the 16 **x**'s represents an average over 16 inner array combinations, or in the case of the marginal effects of the inner array factors, an average over 16 outer array combinations. The lines connect averages for given levels of the appropriate factor.

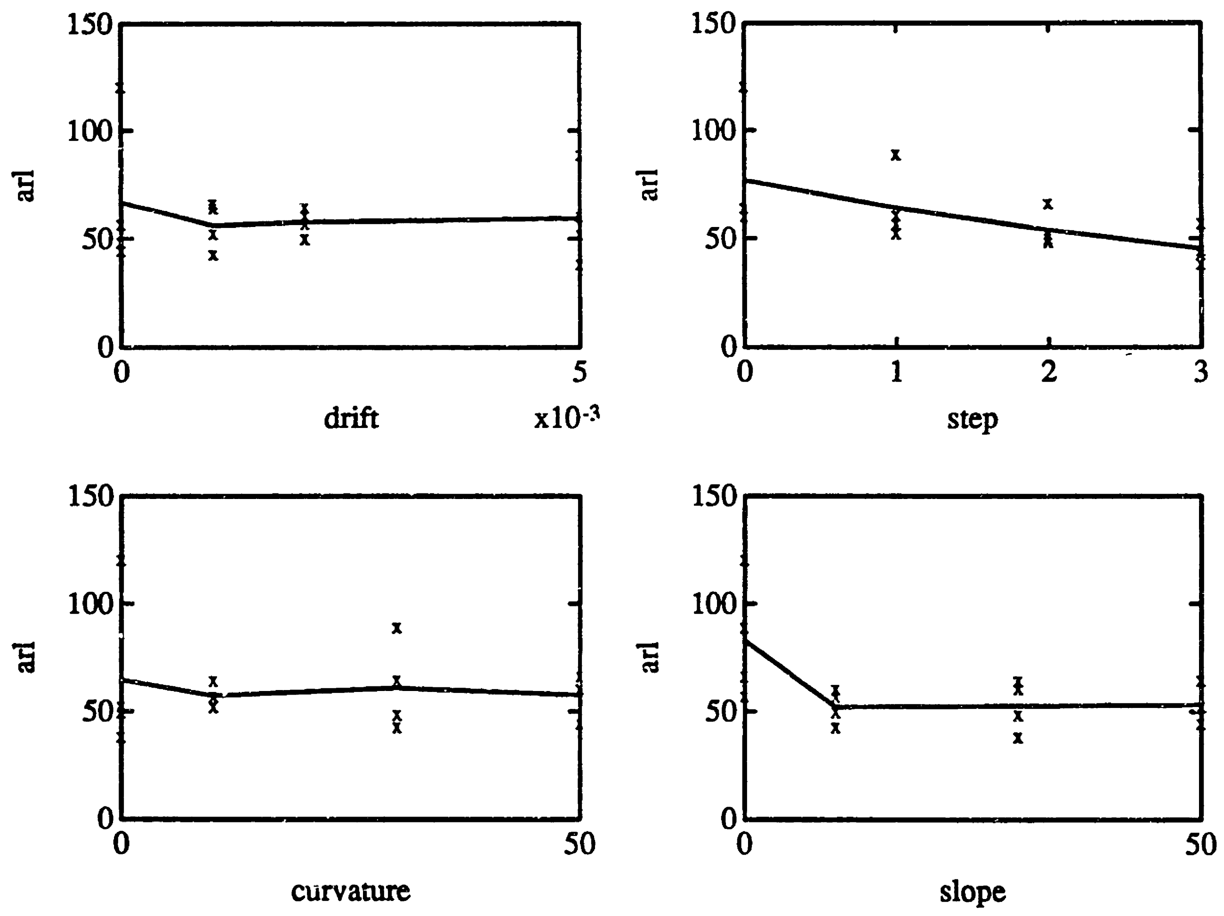


Figure 7-2: The marginal effects of the outer array factors on the average run length, arl.

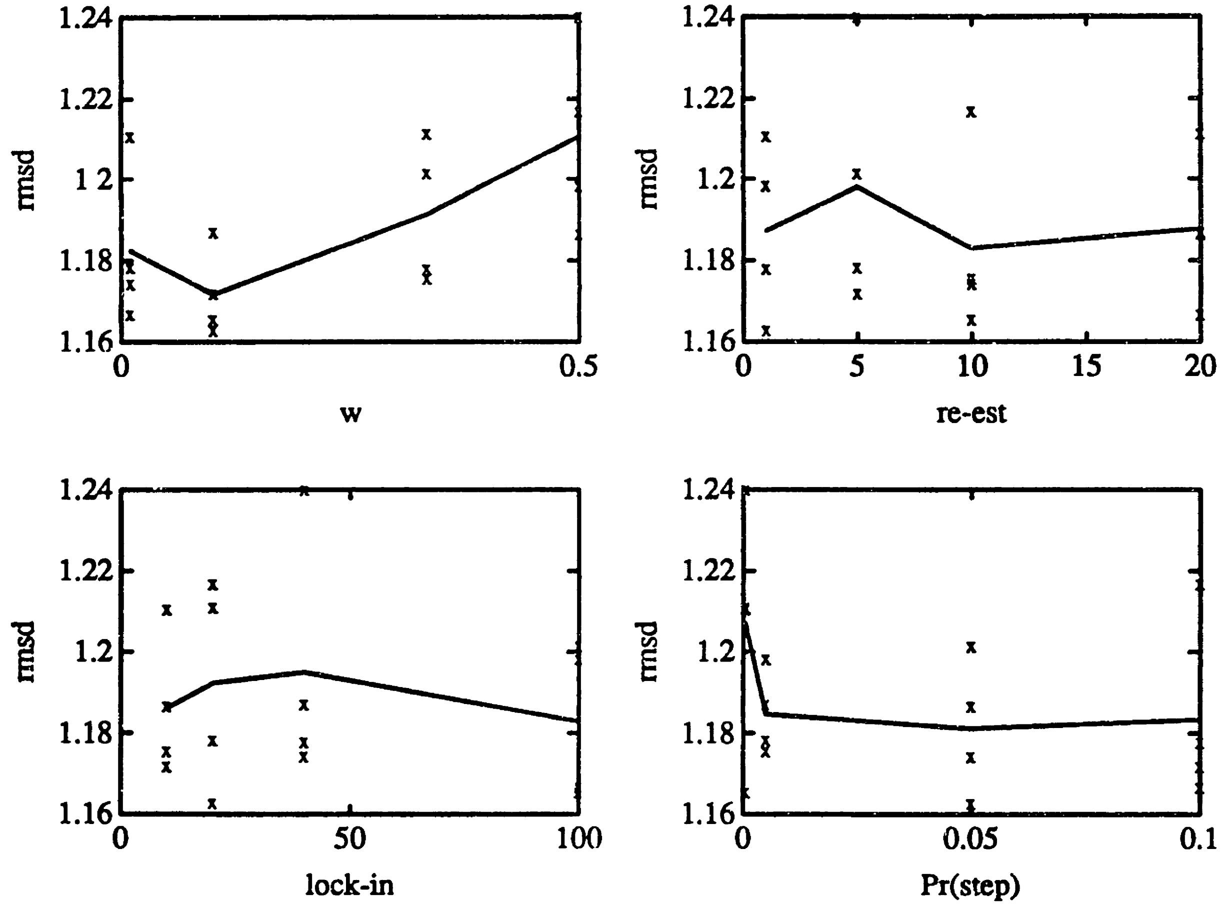


Figure 7-3: The marginal effects of the inner array factors on the rmsd. The factors are the EWMA weight w, the number of times a shift is re-estimated (re-est), the number of runs before a shift is "locked in" (lock-in), and the prior probability of shift (Pr{ step }).

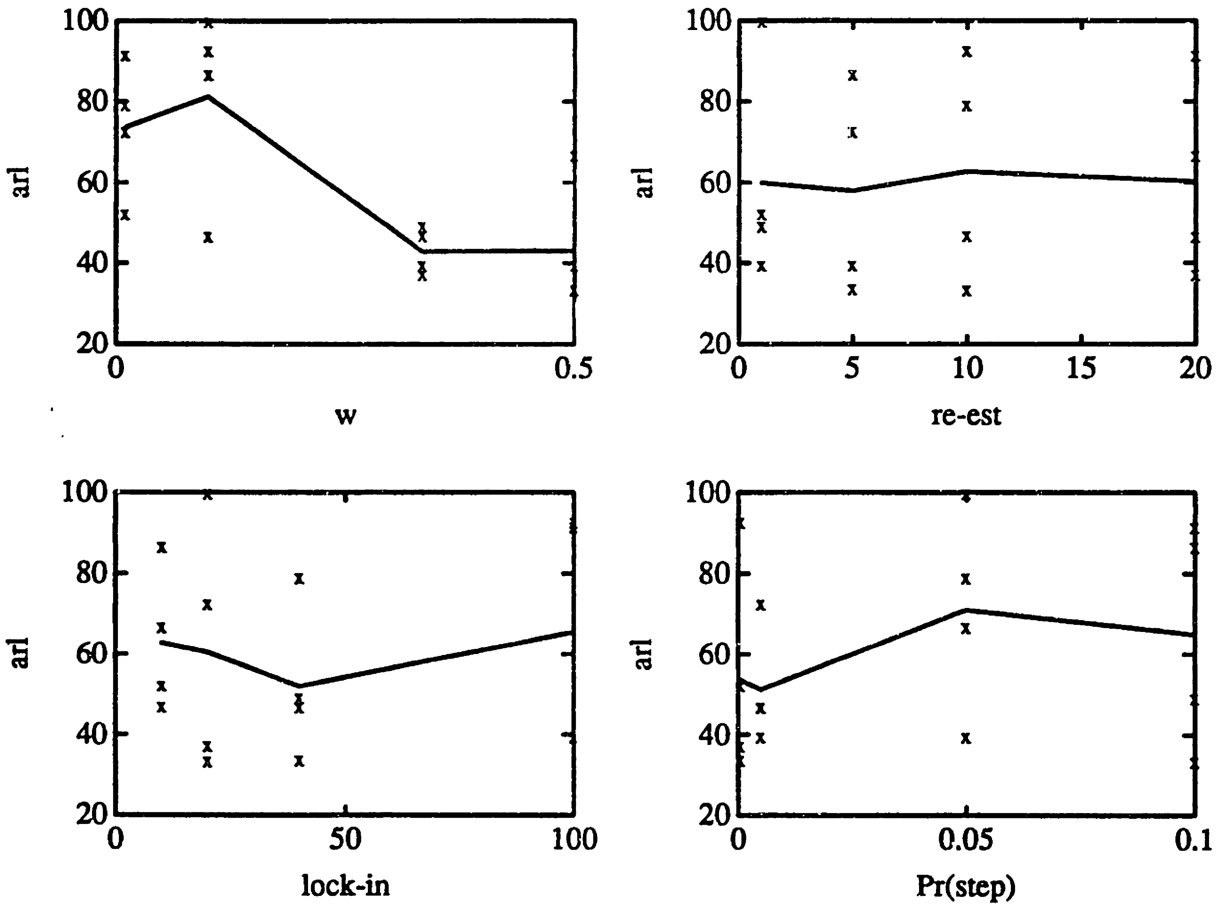


Figure 7-4: The marginal effects of the inner array factors on the arl.

Chapter 8

Conclusions

In this thesis, a control algorithm designed to maintain the performance of a production process that is subject to disturbances which cause the process to move away from target has been presented. This algorithm, referred to as the Run by Run (RbR) controller, combines the methodologies of feedback control and Statistical Process Control (SPC), with feedback control regulating the process, and a generalized version of SPC monitoring the process to detect major disturbances.

The RbR controller was designed to handle the kinds of disturbances most common in practice. At opposite ends of the spectrum of such disturbances are a slow, but steady drift in the process mean on the one hand, and large but occasional shifts in the process mean on the other hand. To deal with those two types of disturbances, the RbR controller has two modes of regulation: the gradual mode, which is designed to compensate for drift, and the rapid mode, whose purpose is to quickly recover the process performance after a shift has occurred. The third major component of the algorithm is a diagnostic mode, which uses generalized SPC to detect the occurrence of shifts. The gradual mode is the default mode, with rapid mode engaged only when the diagnostic mode signals an alarm.

The control strategy used in both gradual and rapid modes is to use measurement data, as it becomes available, to refine a first-order predictive model of the process. This model predicts what the output would be for given input settings. Then, input settings are selected to bring the predicted output on target. In the process model, the first-order coefficients are kept fixed, but the intercept term is updated as an Exponentially Weighted Moving Average (EWMA) of the available process measurements. When in rapid mode, the intercept term is allowed to change more quickly, based on estimates of the magnitude of a recent shift, and the certainty that such a shift really did occur.

Several potential future improvements to the algorithm are explored. These improvements are designed to allow the RbR controller to use second-order models, to better integrate the gradual and rapid modes, and to improve the estimation of shifts in the rapid mode.

The performance of the algorithm was evaluated, both analytically and experimentally. The analytical evaluation consisted of deriving conditions that guarantee the stability of the process output sequence, under different assumptions about the true process behavior. For a first order process with one or more inputs, this condition restricts the range in which the estimated process sensitivities can be, in relation to the true process sensitivities. In particular, the conditions imply that the angle between the vector of estimated process sensitivities **b** and the vector of true process sensitivities β must be smaller than 90 degrees, and the magnitude of **b** "must not be too small" relative to the magnitude of β. However, by using an EWMA statistic that weighs old data more heavily, the minimum magnitude of **b** can be made smaller, at the cost of a decreased speed of convergence. A similar condition was derived for a single input second order process. For a single input first order process with noise and drifting level, the same condition as in the deterministic case was shown to ensure stability in the sense that the expected value of the output sequence converges and its variance is bounded. We also derived expressions which show how the variability of the output of a stable process depends on the ratio of the estimated and true process sensitivities, the magnitude of process drift, and the EWMA weighing scheme. In particular, these expressions show that for any given process parameters, there is a unique EWMA weighing scheme that minimizes the output variation, and that all things being equal, it is safer to err on the side of responding quickly, than to risk being sluggish.

The analytical evaluation applied only to the gradual mode of the RbR controller. The experimental evaluation consisted of a designed experiment, in which a full implementation of the RbR controller was used to control a simulated process. An orthogonal array, with four outer array factors and four inner array factors was used. The outer array factors determined the behavior of the simulated process and the magnitude of disturbances to the process, whereas the inner array factors were parameters of the algorithm.

The results of the experiment show that the as the process becomes more and more different from the idealized model used by the algorithm and as the magnitude of disturbances increases, the performance of the process deteriorates, but for the experimental conditions used, the deterioration was not too severe. The results also suggested that the dependence of the output variation on the EWMA weighing scheme mentioned above carries over to more complicated process behavior.

Several extensions of the work in this thesis are of interest. Perhaps the most important one from a practical point of view is to design and analyze a control strategy that allows for several output variables to be controlled. Other important extensions are to improve and better understand the rapid mode algorithm, to better integrate the gradual and rapid modes, and to allow for the updating of the first-order coefficients on-line.

Finally, an important issue concerns the question of whether to control a process at all. If the behavior of a process were known completely, then in principle, an optimal control scheme could always be designed. But given that knowledge about process behavior is often limited, are there instances where it is better never to take control actions (the SPC "no tweaking" paradigm) ? The design of the RbR controller is based on the belief that even if only a crude model of the process is available, the performance of the process can be improved with feedback control. Lending experimental and analytical support to that belief would be an important contribution.

Bibliography

[Bax90] Robert V. Baxley, Jr. Discussion. *Technometrics*, 32(1):13–16, 1990.

[Bax91] Robert V. Baxley, Jr. Applications of the EWMA for Algorithmic Statistical Process Control. Submitted paper, 1991.

[Ber87] Dimitri Bertsekas. *Dynamic Programming – Stochastic and Deterministic Models.* Prentice-Hall. 1987.

[BJ76] George E. P. Box and Gwilym M. Jenkins. *Time Series Analysis – Forecasting and Control.* Holden-Day, Oakland, CA, 1976.

[BK90] George E. P. Box and Tim Kramer. Statistical Process Control and Automatic Process Control – A Discussion. Technical report, Center for Quality and Productivity Improvement, 1990.

[Dem86] W. E. Deming. *Out of the Crisis.* Massachusetts Institute of Technology Center for Advanced Engineering Study, Cambridge, MA, 1986.

[Dev90] R. L. Devaney. *Chaos, Fractals, and Dynamics.* Addison-Wesley, Menlo Park, CA, 1990.

[Fel91] William H. Fellner. The Riddle of the two Controllers. Manuscript, 1991.

[FHTW90] Frederick W. Faltin, Gerald J. Hahn, William T. Tucker, and Scott Vander Wiel. Algorithmic Statistical Process Control: A Tool for Reducing Product Variability – Some Practical Observations. Unpublished recport, 1990.

[GSHH90] Ruey-Shan Guo, Emanuel Sachs, Albert Hu, and Sungdo Ha. Rapid Process Optimization Using a Run by Run Controller. In *Techcon '90 Rec.*, San Jose, CA, October 1990.

[IS91] Armann Ingolfsson and Emanuel Sachs. Stability and Robustness of an EWMA Controller. To be submitted, 1991.

[Lue79] David G. Luenberger. *Introduction tc Dynamic Systems.* John Wiley & Sons, New York, NY, 1979.

[Lue84] David G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Reading, MA, 1984.

[Mac76] John F. MacGregor. Optimal Choice of Sampling Interval for Process Control. *Technometrics*, 18:151–160, 1976.

[Mac87] John F. MacGregor. Interfaces between Process Control and On-Line Statistical Process Control. *Computing and Systems Technology Division Communications*, 10:9–20, 1987.

[Man69] B. J. Mandel. The Regression Control Chart. *Journal of Quality Technology*, 1(1):1–9, 1969.

[Mit91] Bernd Mittmann, August 1991. Personal communication.

[MLB90] Cleve Moler, John Little, and Steve Bangert. PRO-MATLAB. The MathWorks, Inc., South Natick, MA. 1990.

[Mon85] Douglas C. Montgomery. *Introduction to Statistical Quality Control.* John Wiley & Sons, New York, 1985.

[SGH$^+$91] Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, Albert Hu, Armann Ingolfsson, and Stefan Thomke. *The Run by Run Controller – User's Manual Version 1.0.* Cambridge, MA, 1991.

[SGHH90] Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, and Albert Hu. On-line Process Optimization and Control Using the Sequential Design of Experiments. In *1990 Symposium on VLSI Technology Rec.*, Honolulu, HI, October 1990.

[SGHH91] Emanuel Sachs, Ruey-Shan Guo, Sungdo Ha, and Albert Hu. Process Control System for VLSI Fabrication. *IEEE Transactions on Semiconductor Manufacturing*, 4:134–144, 1991.

[SHI91] Emanuel Sachs, Albert Hu, and Armann Ingolfsson. Run by Run Process Control: Combining SPC and Feedback Control. To be submitted, 1991.

[SHIL91] Emanuel Sachs, Albert Hu, Armann Ingolfsson, and Paul H. Langer. Modeling and Control of an Epitaxial Silicon Deposition Process with Step Disturbances. In *Advanced Semiconductor Manufacturing Conference and Workshop*, Boston, MA, October 1991.

[SPG91] Emanuel Sachs, G. Prueger, and R. Guerrieri. An Equipment Model for Polysilicon LPCVD. To be published in IEEE Transactions on Semiconductor Manufacturing, 1991.

[Tag87] Genichi Taguchi. *System of Experimental Design.* UNIPUB/Kraus International Publications, White Plains, NY, 1987.

[TEH89] Genichi Taguchi, Elsayed A. Elsayed, and Thomas C. Hsiang. *Quality Engineering in Production Systems.* McGraw - Hill, New York, 1989.

[TFW90] William T. Tucker, Frederick W. Faltin, and Scott Vander Wiel. Algorithmic Statistical Process Control: An Elaboration. Unpublished recport, 1990.

[Tuc90] William T. Tucker. Algorithmic Statistical Process Control: Minimum Variance Control with Variable Delays. Unpublished recport, 1990.

[Wes65] Western Electric Company. *Statistical Quality Control Handbook.* Author, Indianapolis, 1965.

[WH89] Mike West and Jeff Harrison. *Bayesian Forecasting and Dynamic Models.* Springer Verlag, New York, 1989.

[WT86] Stanley Wolf and Richard N. Tauber. *Silicon Processing for the VLSI Era.* Lattice Press, Sunset Beach, CA, 1986.

[WTFD90] Scott Vander Wiel, William T. Tucker, Frederick W. Faltin, and Necip Doganaksoy. Algorithmic Statistical Process Control: Concepts and an Application. Unpublished recport, 1990.