# Run-to-Run Control of Static Systems*

Robert L. Kosut, Dick de Roover, Abbas Emami-Naeini, Jon L. Ebert

*SC* Solutions Inc.
3211 Scott Boulevard
Santa Clara, CA 95054 USA
*kosut@scsolutions.com*

**Abstract**

Run-to-run control using static linear models is examined. Conditions are presented for stability, (statistical) performance, and robustness using standard control theory. A simulation example shows the usefulness of the method applied to a rapid thermal oxidation (RTO) process.

## 1 Introduction

The goal of a manufacturing system is to produce multiple copies of the same product, each having attributes within specified tolerances. Product quality attributes are typically determined *after* the product is manufactured, since in most cases sensors are not available which can directly monitor all of these attributes *during* the process. Moreover, during the run, the control variables, or *recipe* variables, are pre-set and held fixed during the run. Different recipes can be selected to produce different products, or similar products but with different attributes.

The run-to-run control problem is to adjust the recipe for the next run based on the results of the previous runs such that the product quality improves and/or yield increases, *i.e.*, more good product is produced. This has proved to be very popular in process control [3].

Previous literature on this topic considers the problem from a statistical point of view (for example, see [1], [2], [3]). This paper describes how the recipe can be adjusted from "run-to-run" using a very simple algorithm based on the attributes of the product produced in the previous run or runs. The algorithm is analyzed under a variety of assumptions on the behavior of the actual process using standard control theory.

*This work is supported by Defense Advanced Research Projects Agency (DARPA)'s applied computational mathematics program under NASA Grant No. NAG-1-1964 and NIST contract No. 50SBNB5C8517.

## 2 Proportional Error Control

Let $t = 1, 2, \ldots$, denote the run number, $r_t \in \mathbf{R}^m$ the vector of recipe variables used during run $t$, $y_t \in \mathbf{R}^n$ the vector of product quality attributes produced at the end of run $t$, and $e_t \in \mathbf{R}^n$ the *normalized* product quality error, whose $i$-th element is defined as,

$$e_t(i) = \frac{y_t(i) - y_{\text{des}}(i)}{y_{\text{tol}}(i)}, \quad i = 1, \ldots, n \tag{1}$$

where $y_{\text{des}}(i)$ is the $i$-th desired product quality attribute, and $y_{\text{tol}}(i)$ is the associated error tolerance. The error, written in vector form, becomes,

$$\begin{aligned} e_t &= R^{-1}(y_t - y_{\text{des}}) \\ R &= \mathbf{diag}\,(y_{\text{tol}}(1), \ldots, y_{\text{tol}}(n)) \end{aligned} \tag{2}$$

The simplest choice for run-to-run control is to correct the previous recipe by an amount proportional to the current error. Thus, for run $t = 1, 2, \ldots$, adjust the recipe according to,

$$\begin{aligned} r_t &= r_{\text{nom}} + u_t \\ u_t &= u_{t-1} - \Gamma e_{t-1}, \quad u_0 = 0 \end{aligned} \tag{3}$$

where $r_{\text{nom}} \in \mathbf{R}^m$ is the nominal recipe, $u_t \in \mathbf{R}^m$ is the correction to the nominal recipe for run $t$, and $\Gamma \in \mathbf{R}^{m \times n}$ is the control design (gain) matrix. It is important to emphasize that (2) together with (3) constitutes the *complete* run-to-run algorithm. Observe also that (3) has the same form as a gradient descent optimization algorithm.

It remains to choose $\Gamma$ and to analyze the algorithm under a variety of assumptions about how $u_t$ effects $e_t$. It will be shown that most of the widely used run-to-run algorithms are in the form of (3) for different choices of the control design (gain) matrix $\Gamma$.

## 3 Static Linear Error System

In this section the run-to-run control (3) is analyzed under the assumption that the actual process can be described by the *static linear error system*.

$$e_t = w_t + G u_t, \quad t = 0, 1, 2, \ldots \tag{4}$$

where $G \in \mathbf{R}^{n \times m}$ is the matrix relating changes in error due to changes in run-to-run control parameters, and $w_t \in \mathbf{R}^n$ is the vector of product quality errors, that would have been produced at the end of run $t$, soley due to the nominal control, *i.e.*,

$$w_t = e_t|_{r_t = r_{\text{nom}}} \tag{5}$$

Note that the run numbering notation used here is different than the standard notation for sampled-data systems where $t = 1, 2, \ldots$ refers to the *sampling instants* (To adhere to the sampled-data standard the error system (2) would read $e_t = w_t + Gu_{t-1}$ which might be confusing). The term *static* refers to the fact that the model (4) has no memory of any inputs other than the previous run, *i.e.*, $G$ is a constant matrix. The effect of a time-varying, dynamic, and/or nonlinear "$G$" will be examined in the future.

Assuming that the actual system is given by (4), application of the run-to-run control (3) results in the error and control (recipe correction) obeying the difference equations,

$$\begin{aligned} e_t &= (I_n - G\Gamma)e_{t-1} + \widetilde{w}_t, \quad e_0 = w_0 \\ u_t &= (I_m - \Gamma G)u_{t-1} - \Gamma w_{t-1}, \quad u_0 = 0 \end{aligned} \tag{6}$$

where $\widetilde{w}_t$ is the run-to-run *variation* in the nominal error,

$$\widetilde{w}_t = w_t - w_{t-1} \tag{7}$$

The fact that the error equation is driven by the run-to-run variation $\widetilde{w}_t$ rather than $w_t$ itself, means that the effect of biases can be eliminated and slow drifts greatly reduced. For example, if

$$w_t = b + ct + v_t \tag{8}$$

where $b \in \mathbf{R}^n$ is a constant bias, $c \in \mathbf{R}^n$ is a constant drift rate, and $v_t \in \mathbf{R}^n$ is a zero-mean random variable, then

$$\widetilde{w}_t = c + v_t - v_{t-1} \tag{9}$$

Clearly $\widetilde{w}_t$ is less potent than $w_t$, particularly if the drift is slow (small $c$) and if the noise variance is small. However, it is important to note that if $v_t$ is a sequence of independent, identically distributed (IID) zero-mean random variables with variance $\sigma^2$, the sequence $v_t - v_{t-1}$, although also zero-mean, has twice the variance, $2\sigma^2$.

### 3.1 Transfer Function Representation

The system variables can also be expressed in transfer function form, with the dependent variable being the backward shift operator $q$. That is, with $u_0 = 0$ and $e_0 = w_0$, for $t = 1, 2, \ldots$ ,

$$\begin{aligned} e_t &= T_{ew}(q)w_t \\ u_t &= T_{uw}(q)w_t \end{aligned} \tag{10}$$

where:

$$\begin{aligned} T_{ew}(q) &= (1-q)\left[I_n - q(I_n - G\Gamma)\right]^{-1} \\ T_{uw}(q) &= -q\left[I_m - q(I_m - \Gamma G)\right]^{-1}\Gamma \\ &= -q\Gamma\left[I_n - q(I_n - G\Gamma)\right]^{-1} \end{aligned} \tag{11}$$

The run-to-run control algorithm (3) can also be expressed as the dynamic error controller,

$$u_t = -K(q)e_t \tag{12}$$

with

$$K(q) = \frac{q}{1-q}\,\Gamma \tag{13}$$

In terms of $K(q)$, the transfer functions become,

$$\begin{aligned} T_{ew}(q) &= [I + GK(q)]^{-1} \\ T_{uw}(q) &= -[I + K(q)G]^{-1} K(q) \\ &= -K(q)\,[I + GK(q)]^{-1} \end{aligned} \tag{14}$$

The transfer function $K(q)$ has a pole at unity which immediately reveals that (3) is an *integral control law*. Hence, as is well known, provided the system is stable, the run-to-run control asymptotically eliminates the effect of bias in $w_t$. This will be made more precise in the sequel.

### 3.2 Stability

It is clear from the above transfer functions that the controlled system is internally stable if and only if the $n$ poles of the transfer function $[I_n - q(I_n - G\Gamma)]^{-1}$ are strictly inside the unit disc. A sufficient condition for this is that the magnitude of the eigenvalues of $I_n - G\Gamma$ are less than one, *i.e.*,

$$|\lambda_i(I_n - G\Gamma)| < 1, \quad i = 1, \ldots, n \tag{15}$$

This is also equivalent to,

$$|1 - \lambda_i(G\Gamma)| < 1, \quad i = 1, \ldots, n \tag{16}$$

Thus, the system is stable if all the eigenvalues of $G\Gamma$ lie strictly inside a unit disc in the complex plane centered at unity on the real axis. Therefore, a sufficient condition for (16) is if,

$$0 < |\lambda_i(G\Gamma)| < 2, \quad \forall i = 1, \ldots, n \tag{17}$$

In the scalar case ($G, \Gamma \in \mathbf{R}$) this condition is both necessary and sufficient because $\lambda_i(G\Gamma) = G\Gamma$. Also, in the special case when the eigenvalues of $G\Gamma$ are all real, then the system is stable if and only if,

$$0 < \lambda_i(G\Gamma) < 2, \quad \forall i = 1, \ldots, n \tag{18}$$

### 3.3 Inverse Control

Suppose that $G$ in (4) is square ($n = m$) and invertible. Assuming $G$ is known, examination of (6) suggests the choice

$$\Gamma = \mu G^{-1} \tag{19}$$

where $\mu \in \mathbf{R}$. The run-to-run algorithm is then,

$$u_t = u_{t-1} - \mu G^{-1} e_{t-1} \tag{20}$$

which together with (4) results in the error and control obeying the difference equations,

$$\begin{aligned} e_t &= (1-\mu) I_n e_{t-1} + \widetilde{w}_t \\ u_t &= (1-\mu) I_m u_{t-1} - \mu G^{-1} w_{t-1} \end{aligned} \tag{21}$$

Observe that this choice of $\Gamma$ diagonalizes the error system. In this case, the system transfer functions are:

$$\begin{aligned} T_{ew}(q) &= \frac{1-q}{1-q(1-\mu)}\, I_n \\ T_{uw}(q) &= -\frac{\mu q}{1-q(1-\mu)}\, G^{-1} \\ K(q) &= \frac{\mu q}{1-q}\, G^{-1} \end{aligned} \tag{22}$$

The system is stable for

$$0 < \mu < 2 \tag{23}$$

In the special case when $\mu = 1$, the difference equations reduce to,

$$\begin{aligned} e_t &= \widetilde{w}_t \\ u_t &= -G^{-1} w_{t-1} \end{aligned} \tag{24}$$

with transfer functions,

$$\begin{aligned} T_{ew}(q) &= (1-q)\, I_n \\ T_{uw}(q) &= -q\, G^{-1} \\ K(q) &= \frac{q}{1-q}\, G^{-1} \end{aligned} \tag{25}$$

### 3.4 Pseudo-inverse control

When the number of controls *m* exceedes the number of errors *n*, set

$$\begin{aligned} \Gamma &= \mu G^{\dagger} \\ G^{\dagger} &= G^T (G G^T)^{-1} \end{aligned} \tag{26}$$

where $G^{\dagger}$ which is the so-called *right psuedo-inverse* of $G$. In this case, the above is the minimum norm $\Gamma$ such that $G\Gamma = \mu I_n$, *i.e.*,

$$\mu G^{\dagger} = \arg\min_{\Gamma} \|\Gamma\| \text{ subject to } G\Gamma = \mu I_n \tag{27}$$

Observe that $(GG^T)^{-1}$ exists if and only if $G$ has full rank, that is, the rank of $G$ is $n$. Also, if $G$ is square, then $\Gamma = \mu G^{-1}$ as in (19). In case $n$ is larger than $m$, a dual result holds using the left pseduo inverse.

With $\Gamma$ from (26), the error and control satisfy the difference equations,

$$\begin{aligned} e_t &= (1-\mu) I_n e_{t-1} + \widetilde{w}_t \\ u_t &= (1-\mu) I_m u_{t-1} - \mu G^{\dagger} w_{t-1} \end{aligned} \tag{28}$$

Equivalently, the system transfer functions are:

$$\begin{aligned} T_{ew}(q) &= \frac{1-q}{1-q(1-\mu)}\, I_n \\ T_{uw}(q) &= -\frac{\mu q}{1-q(1-\mu)}\, G^{\dagger} \end{aligned} \tag{29}$$

The statistical performance measures are identical.

# 4 Statistical Performance Analysis

Suppose that the sequence of uncorrected errors $\{w_1, w_2, \ldots\}$ is a realization of an independent identically distributed (IID) random variable with mean $b \in \mathbf{R}^n$ and covariance $\sigma^2 I_n$, that is,

$$\mathbf{E}\{w_t\} = b, \quad \mathbf{cov}\{w_t\} = \sigma^2 I_n, \quad t = 1, 2, \ldots \tag{30}$$

Assuming the system is stable, *i.e.*, (23) holds, it follows from standard calculations that for all $t \geq 1$,

$$\begin{aligned} \mathbf{E}\{e_t\} &= 0, \qquad \mathbf{cov}\{e_t\} = (\gamma_e \sigma)^2\, I_n \\ \mathbf{E}\{u_t\} &= -G^{-1} b, \qquad \mathbf{cov}\{u_t\} = (\gamma_u \sigma)^2 (G^T G)^{-1} \end{aligned} \tag{31}$$

where $(\gamma_e, \gamma_u)$ depend on $\mu$, *i.e.*, for $0 < \mu < 2$,

$$\begin{aligned} \gamma_e &= \sqrt{\frac{2}{2-\mu}} \\ \gamma_u &= \sqrt{\frac{\mu}{2-\mu}} \end{aligned} \tag{32}$$

Given the above results, RMS statistics will be used to evaluate performance.

The following measures of performance are based on the RMS of individual elements of the error and control sequences. The indices $i, j$ refer to the $i$-th element of the error vector and the $j$-th element of the control.

- The RMS values of the error with and without run-to-run control:

$$\mathbf{rms}\{w_t(i)\}, \quad \mathbf{rms}\{e_t(i)\}, \quad i = 1, \ldots, n \tag{33}$$

- The ratio of RMS control variation about its expected value:

$$\eta_u(j) = \frac{\mathbf{rms}\{u_t(j) - \mathbf{E}\{u_t(j)\}\}}{|\mathbf{E}\{u_t(j)\}|}, \quad j = 1, \ldots, m \tag{34}$$

Another important measure of performance is the system time constant,

$$t_{tc} = -1/\log|1-\mu| \tag{35}$$

Using (30), (31), and (32) gives:

$$\begin{aligned} \mathbf{rms}\{w_t(i)\} &= \sqrt{b(i)^2 + \sigma^2}, \quad i = 1, \ldots, n \\ \mathbf{rms}\{e_t(i)\} &= \sqrt{\frac{2\sigma^2}{2-\mu}}, \quad i = 1, \ldots, n \\ \eta_u(j) &= \left( \frac{\mu}{2-\mu} \frac{(G^T G)^{-1}_{jj}}{(G^{-1} b/\sigma)^2_j} \right)^{1/2}, j = 1, \ldots, m \end{aligned} \tag{36}$$

In general, as $\mu \to 0$, $\mathbf{rms}\{e\} \to \sigma$, which is the smallest possible RMS error, but the time constant $t_{tc} \to \infty$.

In the scalar case, $n = m = 1$, and hence,

$$\begin{aligned} \mathbf{rms}\{w_t\} &= \sqrt{b^2 + \sigma^2} \\ \mathbf{rms}\{e_t\} &= \sqrt{\frac{2\sigma^2}{2-\mu}} \end{aligned} \tag{37}$$

and

$$\eta_u = \frac{1}{|b/\sigma|} \left( \frac{\mu}{2-\mu} \right)^{1/2} \tag{38}$$

As an example, if $|b| = 2\sigma$ then $\mathbf{rms}\{w\} = 2.2\sigma$ and:

$$\begin{array}{llll} \mu = 1 & \Longrightarrow & \mathbf{rms}\{e\} = 1.4\sigma & \eta_u = .5 \quad t_{tc} = 0 \\ \mu = .5 & \Longrightarrow & \mathbf{rms}\{e\} = 1.15\sigma & \eta_u = .3 \quad t_{tc} = 1.44 \\ \mu = .1 & \Longrightarrow & \mathbf{rms}\{e\} = 1.05\sigma & \eta_u = .2 \quad t_{tc} = 9.5 \end{array}$$

Although the RMS error can be reduced from $2.2\sigma$ to $1.05\sigma$ with $\mu = .1$, the time constant is then about 10 runs. Thus, the improvement would not be realized for at least 3 time constants, or about 30 runs. A better compromise would be $\mu = .5$ which gives a similar RMS reduction but with a time constant of 1.44 runs.

Observe that even for $|b|$ as small as $.5\sigma$, it is still possible to reduce the uncorrected RMS error from $1.20\sigma$ to $1.05\sigma$ by setting $\mu = .1$. Again, this will only be realized after at least 30 runs.

## 5 Robust Stability and Performance

Assume now that $G \in \mathbf{R}^{n\times m}$, $m > n$, and has full rank. Let $\widehat{G} \in \mathbf{R}^{n\times m}$ be a full rank estimate of $G$ and let $\Gamma$ be chosen proportional to the approximate pseudo-inverse of the estimate $\widehat{G}$,

$$\begin{aligned} \Gamma &= \mu \widehat{G}^{\dagger} \\ \widehat{G}^{\dagger} &= \widehat{G}^{T}(\widehat{G}\widehat{G}^{T})^{-1} \end{aligned} \tag{39}$$

Suppose that

$$G = (I_n + \Delta)\widehat{G} \tag{40}$$

where $\Delta \in \mathbf{R}^{n\times n}$ represents the *multiplicative uncertainty* between $G$ and the estimate $\widehat{G}$. It is easily verified that the run-to-run control (3) with $Q$ given by (26) results in the error difference equation,

$$e_t = -\Delta e_{t-1} + \widetilde{w}_t \tag{41}$$

The closed-loop transfer functions are now:

$$T_{ew}(q) = (1-q)\,(I_n + q\Delta)^{-1} \tag{42}$$

$$T_{uw}(q) = -q\widehat{G}^{T}(\widehat{G}\widehat{G}^{T})^{-1}\,(I_n + q\Delta)^{-1} \tag{43}$$

The system is stable if and only if the magnitude of the eigenvalues of $\Delta$ are strictly less than unity, *i.e.*,

$$|\lambda_i(\Delta)| < 1, \quad i = 1, \ldots, n \tag{44}$$

Since (41) is a linear difference equation, the solution for $e_t,\ t \geq 0$ can be written explicitly as,

$$e_t = (-\Delta)^t e_0 + \sum_{\tau=0}^{t} (-\Delta)^{t-\tau}\widetilde{w}_\tau \tag{45}$$

and is bounded for all bounded $e_0$ and $w_t$ if (44) holds.

A sufficient condition for stability is that

$$\|\Delta\| \leq \delta < 1 \tag{46}$$

A repeated application of the triangle inequality to (45) together with (46) results in,

$$\|e_t\| \leq \delta^t \|e_0\| + \left(\frac{1-\delta^t}{1-\delta}\right) \max_\tau \|\widetilde{w}_\tau\| \tag{47}$$

Since it is always the case that $\|e_0\| \leq \max_\tau \|w_\tau\|$, it follows that $\|e_t\|$ is bounded by,

$$\begin{aligned} \|e_t\| &\leq \delta^t \max_\tau \|w_\tau\| + \left(\frac{1-\delta^t}{1-\delta}\right) \max_\tau \|\widetilde{w}_\tau\| \\ &\to (1-\delta)^{-1} \max_\tau \|\widetilde{w}_\tau\|, \quad \text{as } t \to \infty \end{aligned} \tag{48}$$

Hence, the performance ratio is bounded by,

$$\begin{aligned} \eta_t &\leq \delta^t + \left(\frac{1-\delta^t}{1-\delta}\right)\left(\frac{\max_\tau \|\widetilde{w}_\tau\|}{\max_\tau \|w_\tau\|}\right) \\ &\rightarrow (1-\delta)^{-1}\left(\frac{\max_\tau \|\widetilde{w}_\tau\|}{\max_\tau \|w_\tau\|}\right), \quad \text{as } t \rightarrow \infty \end{aligned} \tag{49}$$

Observe that the bound on the (asymptotic) performance improvement ratio is reduced from the ideal by $(1-\delta)$.

# 6 Simulation Example

In this section the run-to-run control algorithm is applied to control a rapid thermal oxidation (RTO) process. RTO is a part of Rapid Thermal Processing (RTP) for the processing of silicon dielectrics. The simulation results in this paper are based on a previously derived physical model of a generic RTP system [4], extended with a simplified oxidation model.

The generic RTP system geometry is shown in Figure 1, which is representative of commercial RTP systems. The system consists of five independently powered lamps (inputs) near the top wall that form axi-symmetric rings at radii $r_1,\ldots,r_5$. The walls of the chamber are highly reflective (95%) and water-cooled. A thick quartz window (6.35 mm) and a thinner quartz showerhead (1mm) transmit radiation from the hot lamps at wavelengths shorter than approximately 4$\mu$m, but are opaque to radiation at longer wavelength. The silicon wafer and guard ring are heated by this short wavelength lamp radiation. A physical model of this nonlinear system was constructed that predicts the dynamic temperature response. Details of this model are described in [4]. Five evenly distributed temperature sensors form the measured outputs of this system.

For this example we used a very simple oxidation model, which was taken from [5] . We assume that the *oxidation rate*, $R_{ox}$, is given by:

$$R_{ox} = R_o \exp^{\frac{-E_a}{kT}} \quad [\mathring{A}/\text{second}],$$

with $R_o = 4.60\text{e}5$ $\mathring{A}$/second the oxidation rate constant, $E_a = 1.44$ eV the activation energy, and $k = 8.62\text{e-}5$ eV/K the Boltzmann constant. The thickness of the silicon film, $t_{ox}$, is obtained by integrating the rate w.r.t. time.

In reference [6] temperature control design was performed for the generic RTP model, showing tight temperature control using feedback and feedforward. Figure 2(a) shows a typical reference temperature profile for an oxidation process: heating a wafer from room temperature (25° C) to 1050° C in about 10 seconds, then soak the wafer for about 30 seconds, and finally let the wafer cool-off and take it out. This figure also shows the measured output response, while Figure 2(b) shows the corresponding control input. Figure 2(c) and (d) show the wafer temperature non-uniformity and the oxidation film thickness of 21 wafer

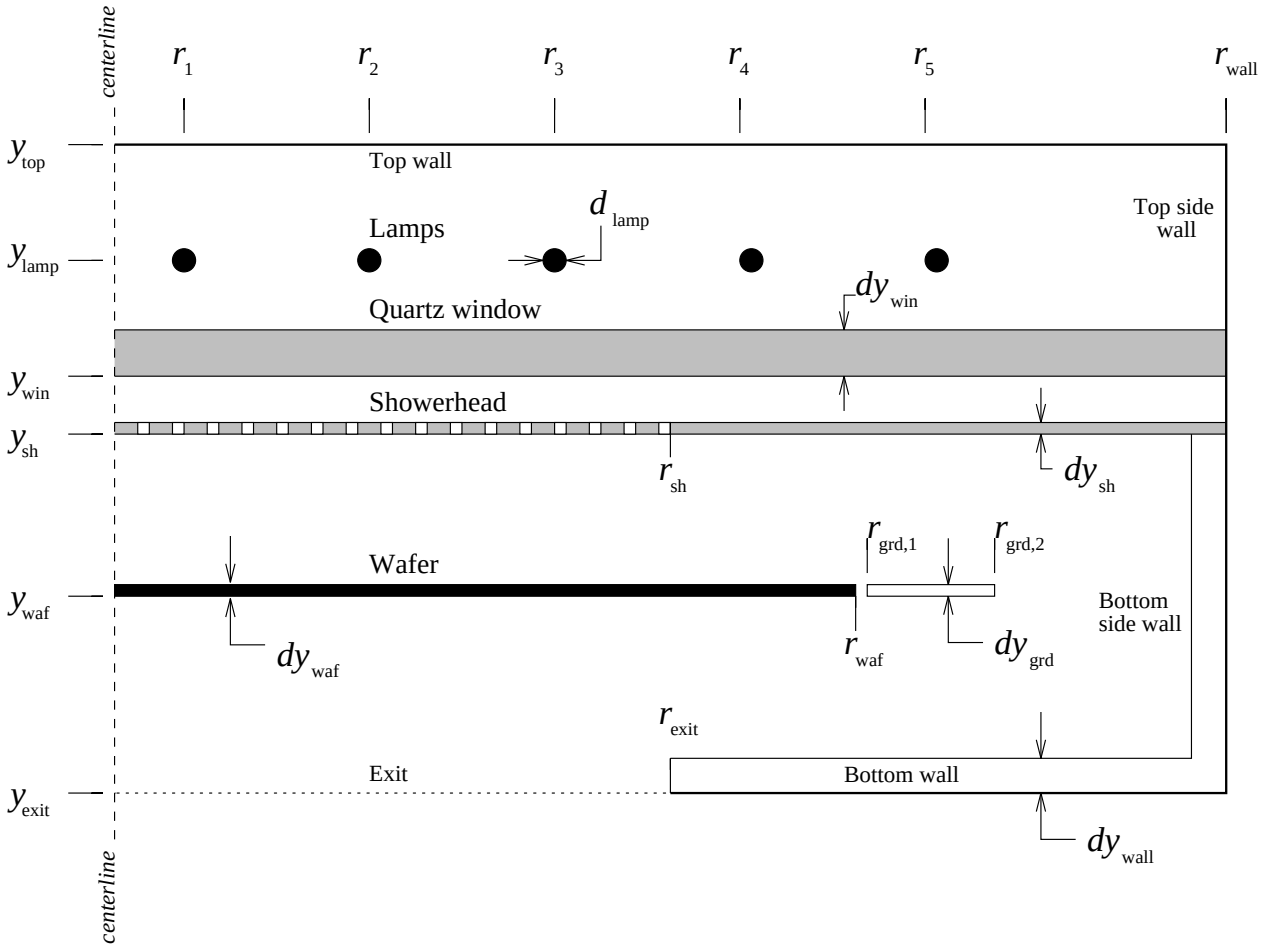


Figure 1: Schematic of the Generic RTP Chamber.

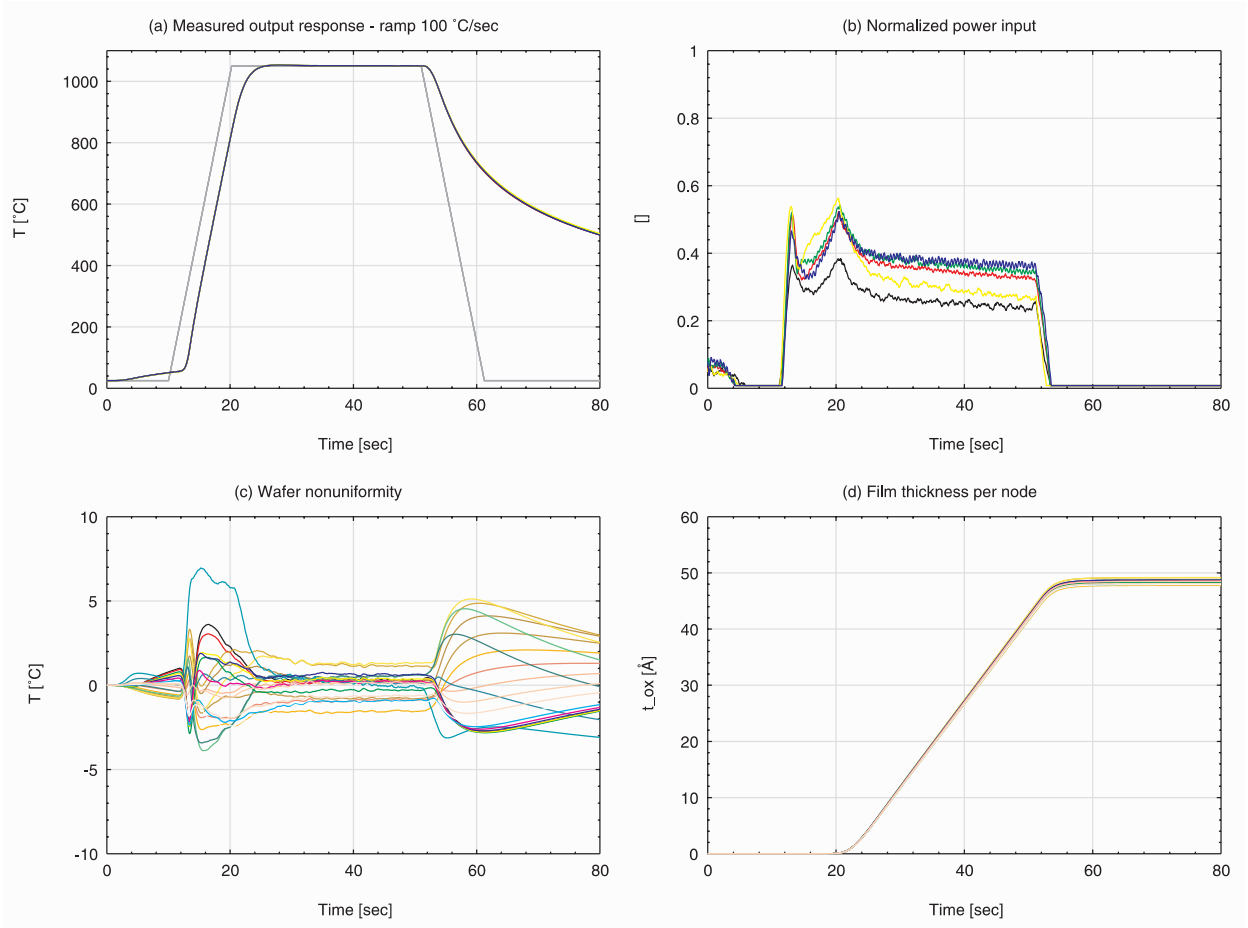


Figure 2: Simulated RTP tracking response; (a) reference ramp $r$ and measured wafer temperature $y$; (b) power input $u$ to RTP system, normalized to 1; (c) wafer temperature non-uniformity for 21 nodes on wafer; each line represents the distance from the average wafer temperature; (d) oxide thickness for each wafer node.

nodes, respectively. Note that the oxidation growth is negligible for temperatures below 800° C, hence the film thickness after 80 seconds can be considered to be the final thickness.

For run-to-run control we choose the film thickness of the 21 wafer nodes at the end of the oxidation process (*i.e.* at $t = 80$ seconds) as the vector of product quality attributes. The 5 soak-temperature setpoints with nominal values equal to 1050° C were chosen as the vector of recipe variables. We assumed a desired film thickness of 50 $\mathring{A}$ for all wafer nodes, and defined the product quality error as:

$$e(i) \doteq 50 - t_{ox}(i) \quad [\mathring{A}], \quad i = 1, \ldots, 21.$$

As the number of product quality attributes (n = 21) is greater than the number of recipe variables (m = 5), we computed the *left* pseudo inverse: $G^{\dagger} = (G^T G)^{-1} G^T$, and set the controller $\Gamma = \mu G^{\dagger}$ for different values

of $\mu$. With this gain matrix we implemented the run-to-run algorithm as:

$$\begin{aligned} e_t &= f(u_t, w_t), \quad w_t \in N(1,1), \quad t = 1, \ldots, 60 \\ u_t &= \begin{cases} 0, & t = 1, \ldots, 20 \\ u_{t-1} - \Gamma e_{t-1}, & t = 21, \ldots, 60 \end{cases} \end{aligned} \tag{50}$$

with $f(u_t, w_t)$ a nonlinear function relating $u_t$ and $w_t$ to $e_t$. Thus, for the first 20 runs, no run-to-run control is applied.

Figure 3(a) shows the oxide thickness before (+) and after (*) run-to-run control, together with the

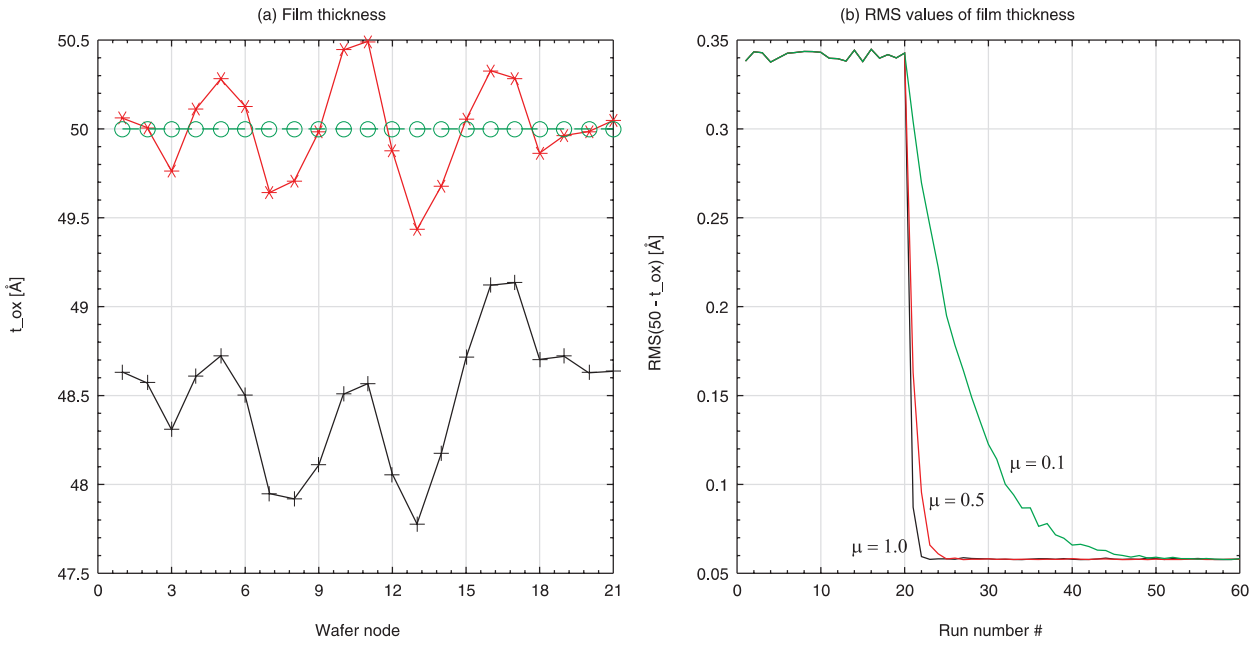


Figure 3: (a) Film thickness for each wafer node before (+) and after (*) run-to-run control, together with desired thickness (o); (b) RMS values of error from desired thickness for different values of $\mu$.

desired thickness (o) of 50 Å. The added value of run-to-run control is clearly visible: the bias w.r.t. the desired thickness has been removed, and the peak-to-peak error has been reduced. Note that it is impossible to reduce the error from the desired thickness to zero for all 21 nodes, as only 5 recipe variables are available. Figure 3(b) shows the RMS value of the error from the desired thickness for different values of $\mu$, with RMS defined as:

$$\text{RMS} \doteq \frac{1}{21}\sqrt{\sum_{i=1}^{21}(50 - t_{ox}(i))^2}.$$

For $\mu = 1$, the run-to-run algorithm converges after 2 runs, whereas for $\mu = 0.1$ it takes about 30 runs. Note that the final RMS value of approximately 0.058 is independent of the value of $\mu$, but depends on the unequal number of recipe variables w.r.t. product quality variables.

Figure 4 shows the effect of different values of $\mu$ at different wafer nodes. This figure shows the trade-off

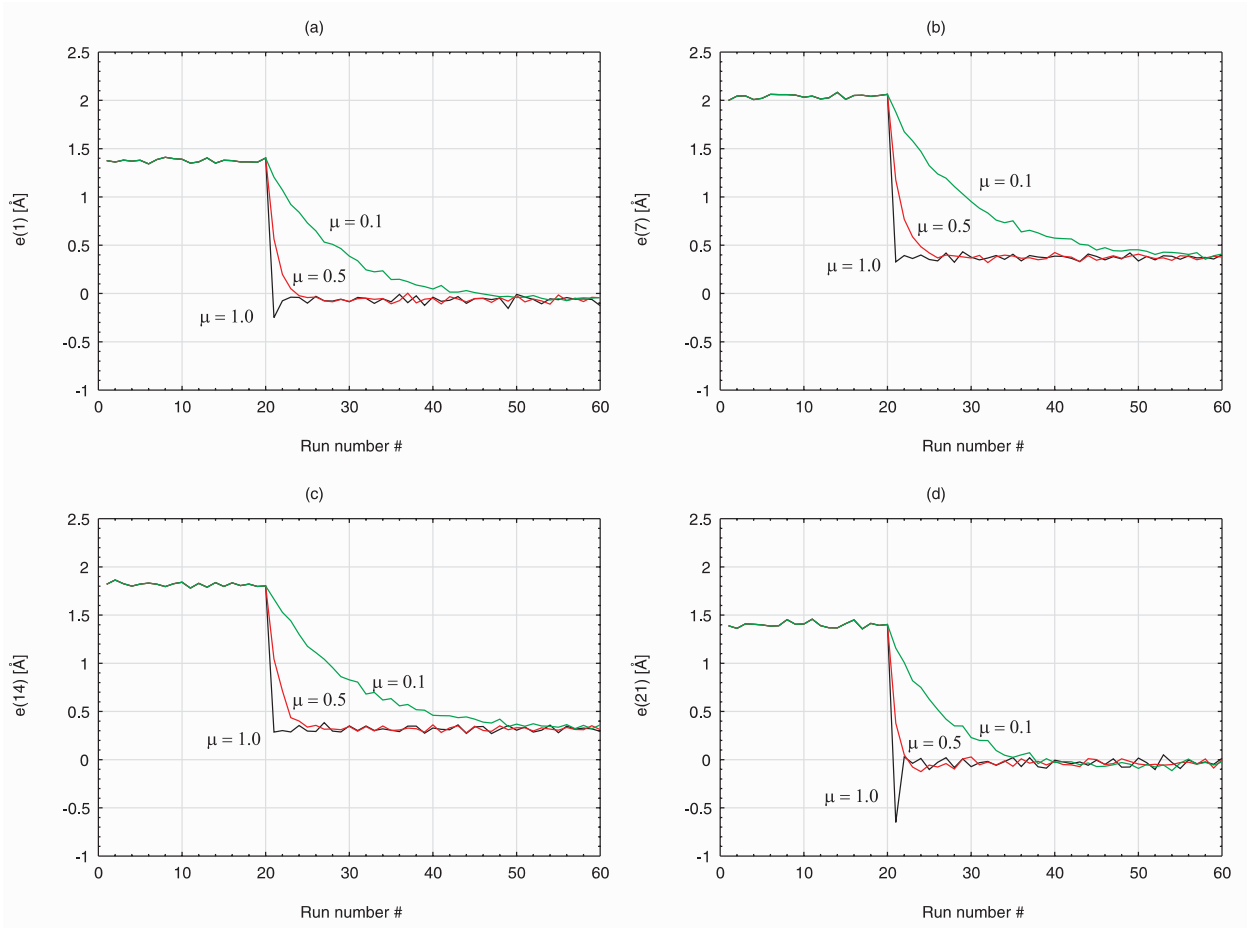


Figure 4: Error from desired film thickness for different wafer nodes and different values of $\mu$: (a) center node; (b) 7th node; (c) 14th node; (d) edge node.

between fast convergence and reduced variance: decreasing $\mu$ gives reduced variance after convergence, but takes more runs.

## 7 Conclusions

We have analyzed a simple run-to-run controller for static systems that can be viewed as a special case of standard integral control. Stability and statistical performance results were developed. The results were applied to an RTP example, showing the usefulness of the algorithm by enforcing convergence of wafer oxide thickness to a desired thickness.

## References


[1] Boning, D., et al.“Run by Run Control of Chemical Mechanical Polishing,” *IEEE Trans. on Components, Packaging, and Manufacturing Technology*, Vol. 19, No. 4, pp. 307-314, Oct. 1996.

[2] Sachs, E., A. Hu, A. Ingolfsson, “Run by Run Processs Control: Combining SPC and Feedback Control,” *IEEE Trans. Semi. Manu.*, Oct. 1991.

[3] Butler, S. W., J. Stefani, “Application of Predictive Corrector Control to Polysilicon Gate Etching,” *Proc. American Contr. Conf.*, June 1993.

[4] Ebert, J. L., A. Emami-Naeini, R. L. Kosut, “Thermal Modeling of Rapid Thermal Processing Systems,” *3rd International Rapid Thermal Processing Conference*, R. B. Fair, B. Lojek (eds), pp. 343-355, Amsterdam, The Netherlands, 1995.

[5] Nulman, J., “Rapid Thermal Processing with Reactive Gases,” in *Reduced Thermal Processing for ULSI*, Ed. R. A. Levy, NATO ASI Series, Plenum, 1989.

[6] Roover, D. de, A. Emami-Naeini, J. L. Ebert, S. Ghosal, G. W. van der Linden, “ Model-Based Control of Fast-Ramp RTP Systems,” *6th International Rapid Thermal Processing Conference*, Kyoto, Japan, 1998.