Designing Response Surface Model Based Run by Run Controllers: A New Approach*

John S. Baras† and Nital S. Patel
Department of Electrical Engineering and Institute for Systems Research
University of Maryland
College Park, MD 20742
baras@src.umd.edu nsp@src.umd.edu

Abstract

This paper presents a framework for carrying out robust Run by Run (RbR) control via a set-theoretic approach. In particular the RbR controller developed tries to minimize the worst case performance of the plant. This gives us a methodology to handle uncertainty. An interesting consequence of using the set valued approach is that now we can relax the assumptions made on the statistics of the noise. Hence, we can also deal with non-Gaussian and correlated noise. We provide results comparing the performance of the controller to a recursive least squares based controller.

1. Introduction

Recently, there has been a strong interest in RbR control in the semiconductor industry. With device tolerances shrinking, it becomes necessary to squeeze maximum performance out of existing equipment. A further advantage of the RbR control framework, is that it enables automatic recipie generation to meet different targets, and also aids in the recovery of the process after a large disturbance. In this paper, we present a worst case framework for carrying out RbR control. The advantage of this approach is its ability to handle uncertainty. This is useful in cases when we do not have confidence in our models, such as after a sudden change in process characteristics. Furthermore, the set theoretic approach followed allows us to relax assumptions on the statistics of the noise. Towards the end of this paper, we present an example where the controller successfully handles both correlated and skewed Gaussian noise.

As with any control strategy, some apriori information needs to be available about the process model. What we require is the structure of the map between the recipie and the measured variables. Such maps could be provided by the response surface models (RSM). However, the conceptual development is not restricted to RSM alone. A number of reseachers have successfully employed RSM to the problem of automated recipie generation, process optimization and design, [8],[12],[11],[9],[13],[7],[2].

In addition to the above, we will also need bounds on the process noise, in the sense that the assumed model, with these noise bounds can account for $(1-\alpha)\%$ of the observevations, where α is a very small number. For example, if we choose α to be 0.27, then the model with these noise bounds can account for 99.73% of the observations. The problem of selecting these bounds is similar to the problem of specifying control limits for control charts in statistical process control (SPC). The idea here is, that if we define our process model in this manner, then the RbR controller hardly ever observes process results which are inconsistent with the model, and if the results are in fact inconsistent, then they will also generate an alarm via SPC. Hence, the RbR controller has to be used in conjunction with SPC. However, by carrying out consistency checks on every measurement, the RbR controller can also generate alarms. The influence of these bounds on the performance of the RbR controller is similar to that observed in control charts. For example, if the bounds are chosen to be smaller than what they actually are, the controller will generate an alarm, even if the process is in control. On the other hand if they are too lax, then the controller becomes less sensitive to process variation.

These bounds are linked to the basic process variance (process noise), over which we have no control and the prediction error of the RSM. For example, if we assume that both the model error, and the process noise are normally distributed with (in the single measurement case) variances σ_M^2, and σ_P^2 respectively, then the bounds around the RSM are $\pm 3\sqrt{\sigma_P^2 + \sigma_M^2}$ for $\alpha = 0.27$. An interesting problem in this regard is given the upper and lower control limits of a variable, and the process noise statistics, what order RSM is required to ensure that the combined model prediction error and the process noise

*This work was supported by the National Science Foundation Engineering Research Centers Program: NSFD CDR 8803012
†Martin Marietta Chair in Systems Engineering

will still result in $(1-\alpha)\%$ of the observations falling between the control limits. Furthermore, for quick recipie calculation, we need the smallest number of terms in the RSM as possible. Initially, one can determine this via a designed experiment. However, once the RbR controller is implemented the process noise statistics and/or the error statistics of the RSM may change. Currently, work is underway to implement a strategy for carrying out automated online modifications of the RSM structure.

The rest of the paper is organized as follows. The general conceptual framework is presented in Section 2. Section 3 specializes the framework to polynomial models. This is followed by simulation results in Section 4, and conclusions in Section 5.

2. Conceptual Framework

This section gives the general framework under which we develop the RbR controler. The system is modeled as

$$\begin{array}{rcl} M_{k+1} & \in & \mathcal{F}(M_k), \quad M_0 = \bar{M} \\ y_{k+1} & \in & \mathcal{G}(M_k, u_k) \\ z_{k+1} & = & l(y_{k+1}), \quad k = 0, \ldots \end{array} \tag{1}$$

Here, $y_k \in \boldsymbol{R}^t$ are the variables to be controlled, $z_k \in \boldsymbol{R}$ is the cost which is a function of y_k, $u_k \in \boldsymbol{U} \subset \boldsymbol{R}^m$ is the vector of recipies, and $M_k \in \mathcal{M}$ are disturbance driven states. At this stage, the structure of $\mathcal{G}$, $\mathcal{F}$, l, and M_k has been left undefined, since the conceptual framework is applicable to any such structure. In the next section, we will fix the structure to obtain an implementable solution. Furthermore, note that the recipies are assumed to belong to the set $\boldsymbol{U}$. One could consider $\boldsymbol{U} \equiv \boldsymbol{R}^m$, however in practice this will hardly ever be the case. In fact natural bounds can be placed on $\boldsymbol{U}$ based on the operating limits of the equipment, using engineering judgement, or by forcing $\boldsymbol{U}$ to be the set of recipies over which the RSM is valid.

The aim of the RbR is to minimize the worst case cost on the onset of every run. Although, the cost has been assumed to be scalar, we could have considered a multi-objective problem with the aim of obtaining Pareto-optimal recipies. We could have also penalized changes in the recipie settings by incorporating additional terms in the cost function l, or restricted the maximum allowable change by redefining $\boldsymbol{U}$ at the onset of each run as a function of the previous recipie settings. The only change required would be in the final optimization stage (defined later on in equation 3).

At this time, it becomes necessary to differentiate between what we call (i) nominal disturbances, and (ii) exceptional disturbances. Nominal disturbances represent the changes that the process normally undergoes between runs. An example of this is drift due to equipment aging. We assume that one can bound these and represent them via $\mathcal{F}$. Exceptional disturbances, on the other hand are those that are not represented by our model (1). Their magnitude is in general much larger than that of the nominal disturbances, and they occur infrequently. They can be caused by various reasons, such as maintainence operations e.g. renewal of parts. Furthermore, exceptional disturbances will also flag an error in the consistency check carried out by the RbR controller. The process noise is modeled separately from the above mentioned disturbances, and is included in $\mathcal{G}$.

We now turn to the problem of designing the RbR controller for the system (1). Assuming, we have carried out $j-1$ runs, we have available to us measurements y_k for $k = 0, \ldots, j$, and past recipies u_k for $k = 0, \ldots, j-1$. Based on this, we compute the set of feasible states $\boldsymbol{P}_j$ that the system could be in during run j assuming that we will not encounter an exceptional disturbance. We can carry out this computation recursively as follows: First compute

$$\mathcal{P}_j = \{M \in \boldsymbol{P}_{j-1} : y_j \in \mathcal{G}(M, u_{j-1})\} \tag{2}$$

and then calculate $\boldsymbol{P}_j$ as

$$\boldsymbol{P}_j = \bigcup_{M \in \mathcal{P}_j} \mathcal{F}(M) \tag{3}$$

The initial set $\boldsymbol{P}_0$ could be defined depending on the amount of confidence one has on the initial value of the states $\bar{M}$. Once, we obtain $\boldsymbol{P}_j$, we obtain the recipie u_j^* which solves

$$\min_{u \in \boldsymbol{U}} \max_{M \in \boldsymbol{P}_j} \max_{y \in \mathcal{G}(M,u)} l(y) \tag{4}$$

In the development above, we have ignored the special case when the set $\mathcal{P}_j$ calculated in (2) is empty. In this case the problem is similar to that encountered in SPC, i.e. to determine whether an exceptional disturbance occured, or that y_j is just a bad data point. This situation can be handled in various ways. Two of these are as follows:

1. Assume that an exceptional disturbance occured and reset $\boldsymbol{P}_{j-1} = \mathcal{M}$. Now recompute $\mathcal{P}_j$. In the next section, and during simulations, we adopt this approach.

2. Assume that y_j is a bad data point. In this case set $\mathcal{P}_j = \boldsymbol{P}_{j-1}$. If the process has shifted, this strategy will generate a large number of faults. We check the number of faults generated in a fixed number of runs, and if this exceeds some threshold value, we reset the set of feasible states to $\mathcal{M}$.

If $\mathcal{F}(M) = \{M\}$, i.e. one does not expect the process to change between runs, then

$$\boldsymbol{P}_j \subset \boldsymbol{P}_{j-1}$$

and hence we have a sequence of nested sets. However, if $\mathcal{F}(M)$ is a set, we can no longer guarantee that the sets remain nested. However, informally one may argue that the larger $\boldsymbol{P}_j$ becomes, more of its elements are invalidated by the next observation y_{j+1}, and hence the sequence of sets $\boldsymbol{P}_j$, $j = 0, 1, \ldots$ will ultimately be bounded. Moreover, since for normal operation $\mathcal{F}(M)$ will be a small set, the inflation introduced by it will also be small. While carrying out simulations, we have observed that this is indeed the case, and the sets $\boldsymbol{P}_j$ do infact remain bounded. Work is currently in progress to derive analytic bounds on these sets.

The main difficulty in the general approach is the excessive computational time required to calculate $\boldsymbol{P}_j$ in (3), and for solving the optimization problem with respect to $\boldsymbol{P}_j$ in (4). However, this result may be used off-line to estimate the best guaranteeable performance achievable under the given model assumptions. We now turn to a technique for approximating these sets using ellipsoids. However, before doing so, we need to impose a suitable structure on the models.

3. Polynomial Models and Ellipsoidal Approximations

Considerable simplifications can be obtained in the above developments, is we impose a polynomial structure on the models. An additional advantage is that this is compatible with the models obtained via RSM [1]. In general, since RSMs are obtained for one quantity at a time, we treat the problem of having t outputs as, t single output problems in parallel. For the i^{th} output y_i, we model $\mathcal{G}_i(\theta_i, u)$ as an ellipsoid given by

$$\mathcal{G}_i(\theta_i, u) \triangleq \{y \in \boldsymbol{R} : g_i^{-1}(y - \theta_i^T u)^2 \leq 1\} \qquad (5)$$

where $u = [1\ v_1\ v_2\ \ldots\ v_m]^T$, with v_j, $j = 1, \ldots, m$ being known functions of the recipie settings, and θ_i being the vector of coefficients. In particular, $\theta_i^T u$ represents the RSM for the quantity y_i, and $g_i > 0$ represents the bound on the noise. Also, (5) implies that

$$y_i \in [\theta_i^T u - \sqrt{g_i}\ ,\ \theta_i^T u + \sqrt{g_i}] \qquad (6)$$

This representation can take care of non-symmetric bounds on the noise by adding a bias term to the first element of θ_i.

We now turn to the definition of $\mathcal{F}_i(\theta, u)$, which we also define as an ellipsoid as

$$\mathcal{F}_i(\bar{\theta}) \triangleq \{\theta \in \boldsymbol{R}^m : (\theta - \bar{\theta}\ )^T F_i^{-1} (\theta - \bar{\theta}\ ) \leq 1\}$$

with $F_i \in \boldsymbol{R}^{m \times m}$, $F_i > 0$, i.e. F_i is a positive definite matrix. As an example, for the case of a process with drift only, F_i will be diagonal with very small entries for all the diagonal elements, except the first.

Ellipsoidal algorithms return an ellipsoidal estimate of the feasible parameter set. Specifically, on the onset of run k, we have

$$\boldsymbol{P}_{i,k} = \{\theta_i \in \boldsymbol{R}^m : (\theta_i - \bar{\theta}_{i,k})^T P_{i,k}^{-1} (\theta_i - \bar{\theta}_{i,k}) \leq 1\}$$

where $\bar{\theta}_{i,k}$ is the center of the ellipsoid, and the matrix $P_{i,k} \in \boldsymbol{R}^{m \times m}$, $P_{i,k} > 0$ specifies the size and orientation of the ellipsoid. Various ellipsoidal algorithms exist in the literature, [6],[4],[14]. The algorithm implemented by us, has two stages:

1. First using the optimal volume ellipsoid (OVE) approach of [4], we try to find the minimum volume ellipsoid $\hat{\boldsymbol{P}}_{i,k}$ which bounds the set

   $$\{\theta_i \in \boldsymbol{P}_{i,k-1} : y_k \in \mathcal{G}_i(\theta, u_{k-1})\}$$

   where $\boldsymbol{P}_{i,k-1}$ is the ellipsoidal estimate of θ_i at the onset of run $k-1$.

2. We then inflate $\hat{\boldsymbol{P}}_{i,k}$ by finding the minimal volume ellipsoid which bounds

   $$\bigcup_{\theta \in \hat{\boldsymbol{P}}_{i,k}} \mathcal{F}_i(\theta)$$

   and set that equal to $\boldsymbol{P}_{i,k}$. For this stage, we use a result from [3].

The initial values of $P_{i,0}$ can be fixed depending on the amount of confidence we have on the initial parameter vectors $\bar{\theta}_{i,0}$. Furthermore, if the intersection in stage 1 of the update algorithm returns an empty set, we reset the entries of $P_{i,k-1}$ to large values (in certain cases, we may be able to use our judgement and only modify selected entries) and then repeat stage 1.

Now assuming, we have t outputs, and hence t ellipsoids characterized by $(\bar{\theta}_{i,k}, P_{i,k})$, $i = 1, \ldots, t$, we can pose the final optimization problem (4) as

$$\min_{u \in \hat{\boldsymbol{U}}} \max_{\underline{y}_k \leq y \leq \overline{y}_k} l(y) \qquad (7)$$

where $\hat{\boldsymbol{U}} = \{[u_0\ v]^T : u_0 = 1, v \in \boldsymbol{U}\}$, and $\underline{y}_k$, $\overline{y}_k$ are t dimensional vectors, whose i^{th} components are given by $\bar{\theta}_{i,k}^T u - \sqrt{u^T P_{i,k} u} - \sqrt{g_i}$ and $\bar{\theta}_{i,k}^T u + \sqrt{u^T P_{i,k} u} + \sqrt{g_i}$ respectively. We now force l to be convex with respect to y (such as a quadratic cost function). Then, the inner maximization is achieved at one of the verticies of the *box* defined by $\underline{y}_k$, $\overline{y}_k$ and the optimization algorithm takes advantage of this fact to cut down the complexity.

We mention here the fact, that the OVE algorithm does not propagate the center of the ellipsoid as a weighed recursive least squares (WRLS) estimate. However, there do exist ellipsoidal algorithms which do propagate the center as a WRLS estimate of the model parameters [6],[5]. In these cases, however, the ellipsoid may greatly overbound the feasible set, and hence could yield very conservative recipies. If however, the bound is tight, then one can view (7) as minimizing the cost based on the WRLS estimate with modulation due to the uncertainty in the estimate. However, trying to measure the tightness of the bounds is as hard as solving for the actual feasible set, i.e. the exact problem of the previous section.

Note that, since the feasible set for each output is computed independently of the others, we could sample the different outputs at different rate with only minor modifications to the structure of the RbR controller. For example, if we had only two outputs (y_1,y_2), we could sample y_1 after every run, and y_2 after every two runs. Then we would update $\boldsymbol{P}_1$ as mentioned above. However, for $\boldsymbol{P}_2$, we would do a full update only when we obtain a new measurement, else we will only inflate it as done in step 2 of the update algorithm. This also raises the possibilities of carrying out multi-rate sampling.

4. Simulation Results

In this section, we present some simulation results. We consider four cases: (i) system under steady drift, (ii) a step disturbance, (iii) presence of bad data points, and (iv) correlated noise. the simulations are based on the models for an LPCVD reactor presented in [10]. Here, we limit our concern to the deposition rates on the first and last wafer. We augment the models with drift terms. The models express the deposition rates in terms of deposition temperature T, deposition pressure P, and the silane flow rate Q. They are given by

$$\begin{aligned} \hat{R}_1 &= \exp(c_1 + c_2 \ln P + c_3 T^{-1} + c_4 Q^{-1}) + d_1 \\ \hat{R}_2 &= \hat{R}_1 \left[\frac{1 - S' C_{gs} \hat{R}_1 Q^{-1}}{1 + S' C_{gs} \hat{R}_1 Q^{-1}}\right] + d_2 \end{aligned} \tag{8}$$

with the rates expressed in Å/min, P in mtorr, T in K, and Q in sccm. The parameters are given [10] to be $c_1 = 20.65$, $c_2 = 0.29$, $c_3 = -15189.21$, $c_4 = -47.97$, $S' = 4777.8$, and $C_{gs} = 1.85 \times 10^{-5}$, where we have dropped the units for convenience. d_1, and d_2 represent the drift terms. The measured rates are obtained from the above model by adding a zero mean noise to $\hat{R}_1$ and $\hat{R}_2$. For the first two cases, we take it to be gaussian with variance 9, and for the last case, we filter it to obtain a colored noise. Furthermore, the maximum drift expected between runs is 0.3. This actually represents a shift of σ in 10 runs, and may be too large to be true in practice. However, we choose this value, since it enables us to see the corrective action of the RbR controller in a fewer number of runs. The targets T_1 for R_1, and T_2 for R_2 are fixed at 169.75 Å/min and 141.7 Å/min respectively.

For the purpose of the RbR controller, we work in the ln space. The controller observes $y_1 = \ln R_1$, and $y_2 = \ln R_2$. We now set $u_1 = \ln P$, $u_2 = T^{-1}$, and $u_3 = Q^{-1}$, and define the vector $u = [1\ u_1\ u_2\ u_3]^T$. Assuming, that the RbR controler keeps the process on target, we fix g_1, and g_2 as 0.0025. Note, that this value yields smaller bounds than the actual bounds on the noise, however, due to overbounding by the ellipsoids, no consistency error was generated by the RbR controller. Based on the drift information, we now fix F_1 and F_2 as

$$F_1 = \begin{bmatrix} 3.2 \times 10^{-4} & 0 & 0 & 0 \\ 0 & 10^{-12} & 0 & 0 \\ 0 & 0 & 10^{-12} & 0 \\ 0 & 0 & 0 & 10^{-12} \end{bmatrix}$$

$$F_2 = \begin{bmatrix} 4.7 \times 10^{-4} & 0 & 0 & 0 \\ 0 & 10^{-12} & 0 & 0 \\ 0 & 0 & 10^{-12} & 0 \\ 0 & 0 & 0 & 10^{-12} \end{bmatrix}$$

The initial parameter vectors $\theta_{1,0}$, and $\theta_{2,0}$ are fixed as

$$\theta_{1,0} = \begin{bmatrix} 20.65 \\ 0.29 \\ -15189.21 \\ -47.97 \end{bmatrix} \quad ; \quad \theta_{2,0} = \begin{bmatrix} 16.3509 \\ 0.2177 \\ -10992 \\ -71.774 \end{bmatrix}$$

where $\theta_{2,0}$ is obtained by fitting a polynomial model to the data generated by the system (7). The model is valid for a fixed range of the inputs, namely the experimental design space. Hence, we place bounds on the recipie settings, and these translate to the vector u as follows: $5.67 \leq u_1 \leq 6.33$, $1.053 \times 10^{-3} \leq u_2 \leq 1.1777 \times 10^{-3}$, and $0.003 \leq u_3 \leq 0.01$. We initialize $P_{1,0} = P_{2,0} = 10^{-12} I$, where I is a 4×4 identity matrix. Finally, the cost is expressed as a quadratic function of the measurements as

$$l(y) = w_1 (y_1 - \ln T_1)^2 + w_2 (y_2 - \ln T_2)^2$$

where, $w_1 = (\ln T_1)^2$, and $w_2 = (\ln T_2)^2$. These, weigh the component involving y_1 more than that involving y_2, since the former is less sensitive to error in the deposition rate, due to the non-linearity of the ln transformation. Moreover, one can experiment with different weights, however our results show these weights to be good enough.

(i) Drifting Process

We let both the equations (7) have a drift of -0.3 Å/min between runs. Simulation results for both the controlled and uncontrolled trajectories are displayed in Figure 1. The dash-dot lines give the target and the 3σ noise

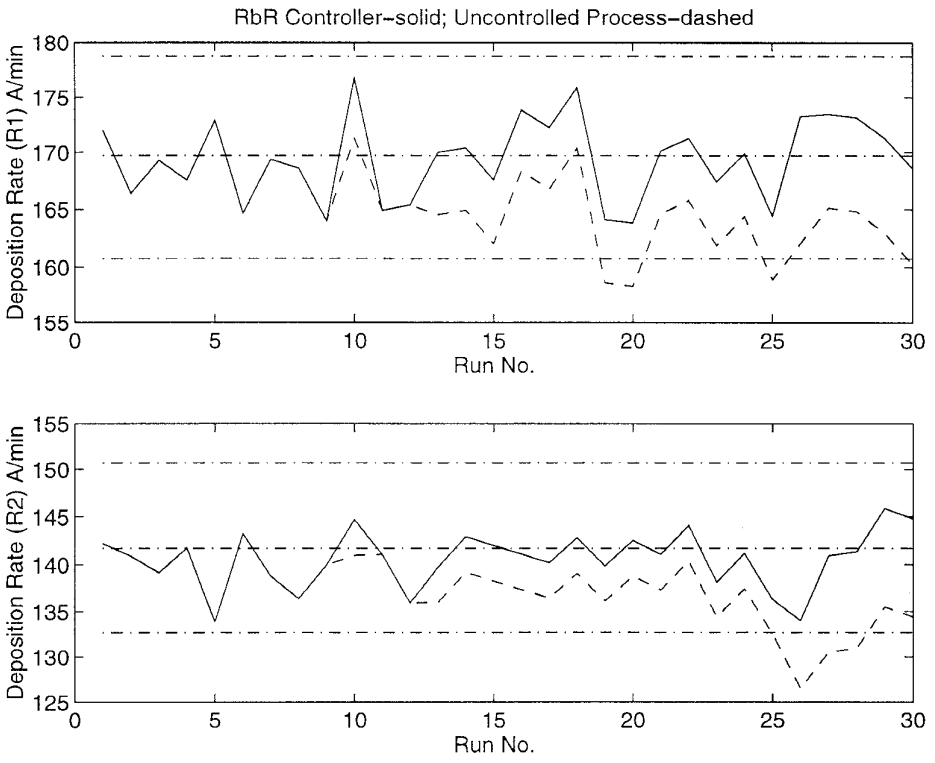

Figure 1: Process under drift.

bounds, where $\sigma = 3$ is the standard deviation of the noise. For comparison, both the controlled and uncontrolled trajectories have been obtained under identical noise conditions.

(ii) Step Disturbance

Here, we change the process parameters after the 3^{rd} run. The parameters are changed to $c_3 = -14600$, $c_4 = -55.97$, $d2 = 11$, and $C_{gs} = 1.57 \times 10^{-5}$. The controller generated a consistency fault after run 3, and we reset P_1, and P_2 to $P_{1,3} = P_{2,3} = 10^6 I$. Figure 2 shows the controlled and uncontrolled trajectories obtained under identical noise conditions. The dash-dot lines represent the target, and the noise limits as in Figure 1.

(iii) Bad Data Points

Two bad data points are generated during the simulation runs. These occur during run 10 for R_1, and run 20 for R_2. The controlled trajectories are shown in Figure 3. The dash-dot lines represent the target, and the noise limits. The simulations show the controller to be minimally affected by bad data points.

(iv) Colored Noise

It should be noted that the noise seen by the controller is in fact skewed due to the ln transformation of the data. We now color the noise via filters, such that the first correlation coefficient (i.e. $E(n_i n_{i-1})/\sigma^2$) is equal to -0.2680 for the noise added to $\hat{R}_1$, and -0.0627 for the noise added to $\hat{R}_2$. The simulation is similar to the step disturbance case, and we present the plot for data from runs 6 to 30 in Figure 4. Here, WCA corresponds the controller designed in this paper. We also plot the output (shown via a dashed line) obtained by using a controller based on the recursive least squares (RLS) estimate of the model coefficients. For purposes of comparison, both the WCA and RLS based controllers were simulated with the process subject to identical noise. The means and standard deviations (STD) of the outputs are given in Table 1. The RbR controller presented in this paper outperforms the RLS based controller in terms of both mean and standard deviation.

5. Conclusions

A worst case approach to RbR control is presented, and we have demonstrated its viability via simulation results. It is able to compensate for drift, and step changes. We

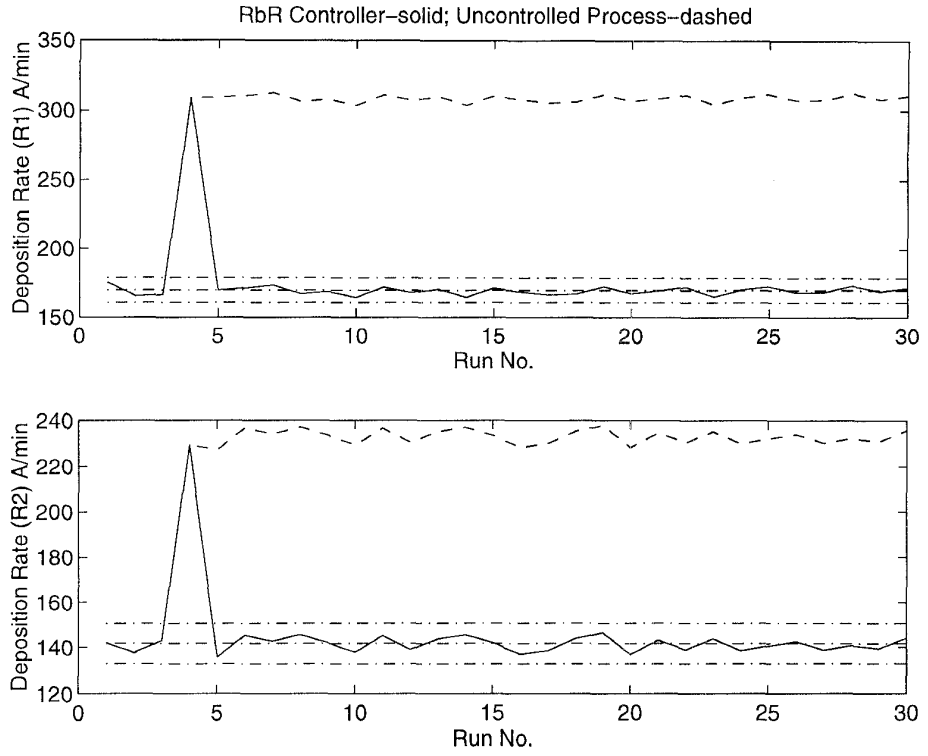

Figure 2: Process with a step disturbance.

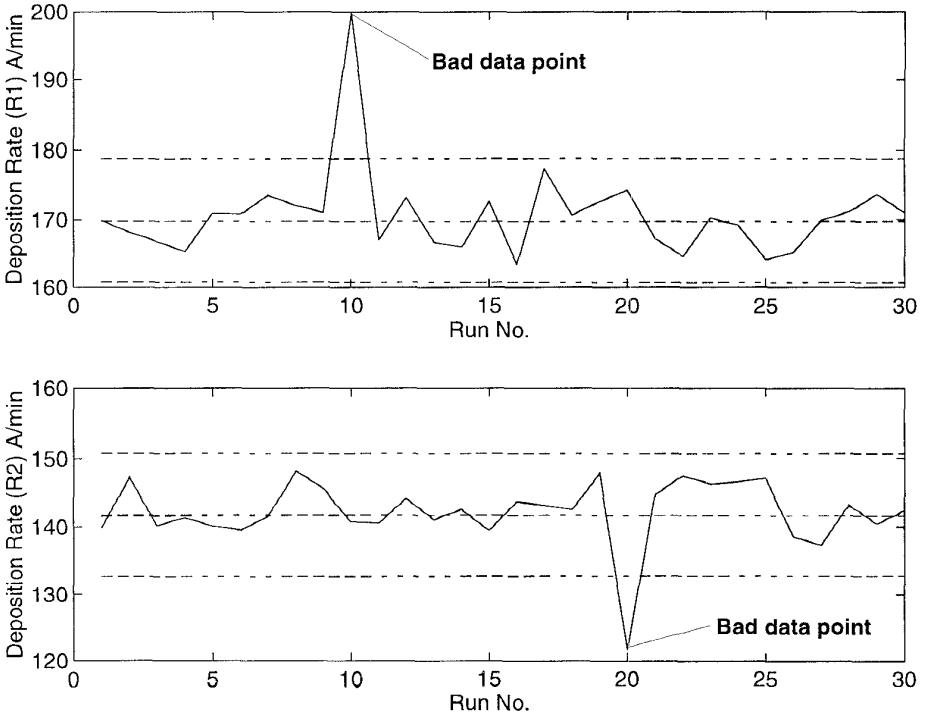

Figure 3: Effect of bad data points.

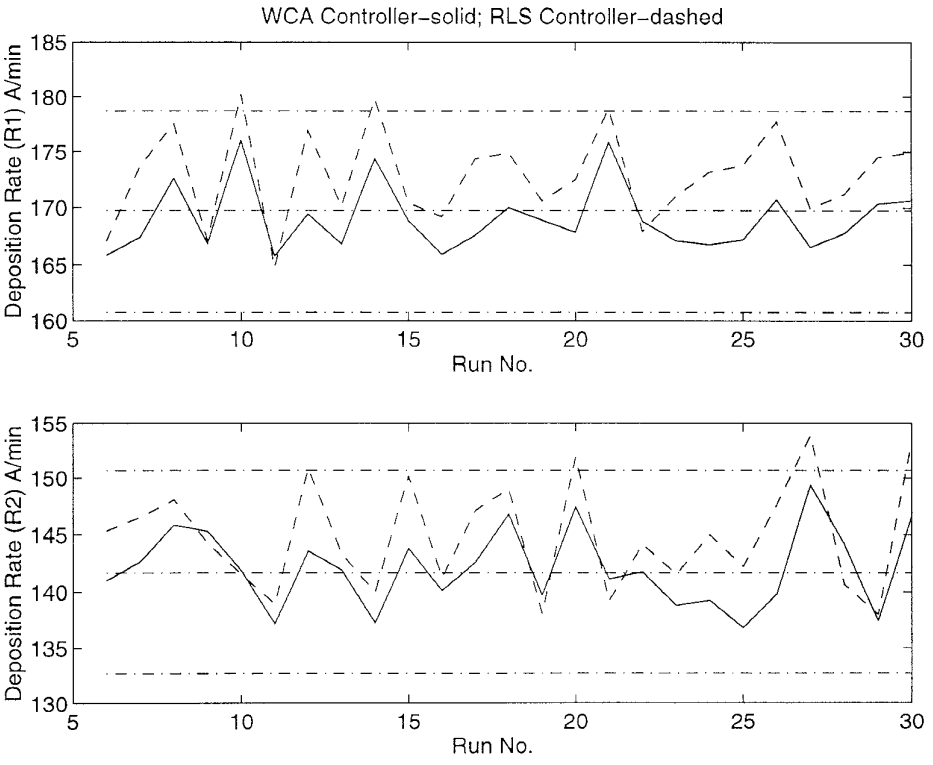

Figure 4: Process with colored noise.

RLS	R_1	mean	172.90
		STD	4.19
	R_2	mean	144.87
		STD	4.81
WCA	R_1	mean	169.04
		STD	2.96
	R_2	mean	142.08
		STD	3.48

Table 1: Comparison of WCA and RLS Controllers

have also demonstrated its robustness to skewed and colored noise.

Work is continuing to develop an expert system based monitor for automated model order changes, and to carry out online tuning of the RbR controller. We are also looking into an application of the controller to an industrial process.

We would also like to point out the fact that level sets in probability obtained from multi-variate normal distributions are in fact ellipsoids. Hence, the ellipsoidal algorithm maybe equivivalent to fitting a normal distribution to the statistics of the estimated process parameters. However, the exact relationship between the two is still an open question. One can also view the ellipsoids as estimating a confidence set for the RSM parameters.

References

[1] G.E.P. Box and N.R. Darper. *Empirical Model-Building and Response Surfaces.* Wiley, New York, 1987.

[2] S.W. Butler and J.A. Stefani. Supervisory run-to-run control of polysilicon gate etch using *in situ* ellipsometry. *IEEE Transactions on Semiconductor Manufacturing*, 7(2):193–201, 1994.

[3] F.L. Chernous'ko. Optimal guaranteed estimates of indeterminacies with the aid of ellipsoids. *Engineering Cybernetics*, 18:1–9, 1980.

[4] M.F. Cheung, S. Yurkovich, and K.M. Passino. An optimal volume ellipsoid algorithm for parameter set estimation. In *Proc. 30th IEEE Conference on Decision and Control*, pages 969–974, 1991.

[5] J.R. Deller, M. Nayeri, and S.F. Odeh. Least-square identification with error bounds for real time signal processing and control. *Proceedings of the IEEE*, 18(6):815–849, 1993.

[6] E. Fogel and Y.F. Huang. On the value of information in system identification-Bounded noise case. *Automatica*, 18(2):229–238, 1982.

[7] G.J. Gaston and A.J. Walton. An integration of simulation and response surface methodology for the optimization of IC processes. *IEEE Transactions on Semiconductor Manufacturing*, 7(1):22–33, 1994.

[8] S. Leang and C.J. Spanos. Statistically based feedback control of photoresist application. In *Proc. IEEE/SEMI Advanced Semiconductor Manufacturing Conference*, pages 185–190, 1991.

[9] S. Leang and C.J. Spanos. Application of feedforward control to a lithography stepper. In *Proc. IEEE/SEMI Int'l Semiconductor Manufacturing Science Symposium*, pages 79–84, 1992.

[10] K-K. Lin and C.J. Spanos. Statistical equipment modeling for VLSI manufacturing: An application for LPCVD. *IEEE Transactions on Semiconductor Manufacturing*, 3(4):216–229, 1990.

[11] P.K. Mozumder and G.G. Barna. Statistical feedback control of a plasma etch process. *IEEE Transactions on Semiconductor Manufacturing*, 7(1):1–11, 1994.

[12] S. Pan, M.T. Reilly, E. Di Fabrizio, Q. Leonard, J.W. Taylor, and F. Cerrina. An optization design method for chemically amplified resist process control. *IEEE Transactions on Semiconductor Manufacturing*, 7(3):325–332, 1994.

[13] J.A. Power, B. Donnellan, A. Mathewson, and W.A. Lane. Relating statistical MOSFET model parameter variabilities to IC manufacturing process fluctuations enabling realistic worst case design. *IEEE Transactions on Semiconductor Manufacturing*, 7(3):306–318, 1994.

[14] F.C. Schweppe. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Transactions on Automatic Control*, 13(1):22–28, 1968.