COMPUTING PARAMETRIC YIELD ADAPTIVELY USING LOCAL LINEAR MODELS[*]

Mien Li *Linda Milor*

Electrical Engineering Department and Institute for System Research
University of Maryland at College Park
mienli@src.umd.edu
milor@src.umd.edu

Abstract— A divide-and-conquer algorithm for computing the parametric yield of large analog circuits is presented. The algorithm targets applications whose performance spreads could be highly nonlinear functions of a large numbers of stochastic process disturbances, and therefore can not easily be modeled by traditional response surface methods. This work addresses difficulties with modeling by adaptively constructing the model piece by piece, namely, by efficiently and recursively partitioning the disturbance space into several regions, each of which is then modeled by a local linear model. Local linear models are used because they are less sensitive to dimension than polynomial models. Moreover, the resulting model can be made to be more accurate in some regions compared to others. The number of simulations required in statistical modeling can therefore be reduced since only critical regions, which define the boundary of the feasible region in the space of process disturbances, are modeled highly accurately. The resulting models are then used as cheap surrogates for circuit simulation in Monte Carlo estimation of the parametric yield. Examples indicate the efficiency and accuracy of this approach.

1. Introduction

Circuit yield depends on the immunity of a design to both defects and process fluctuations. This paper is specifically concerned with yield loss caused by stochastic disturbances in the manufacturing process, refered to as *parametric yield*. As we move towards deep submicron processes, these inherent stochastic disturbances are becoming more difficult to characterize and model. And moreover, as designs become more complex, the number of process disturbances which cause significant circuit performance spreads is increasing. With the current high cost of submicron fabrication processes, the need for high yielding designs with shorter product development times drives the need for an efficient and accurate method for estimating the parametric yield prior to manufacture.

In order to compute parametric yield, it is necessary to identify a set of disturbances that characterize process fluctuations and the specifications on circuit performances which must be satisfied. Many papers have focused on process characterization for both analog and digital circuits [1, 2]. This paper assumes that circuit specifications are known and potentially critical process disturbances have been identified and focuses on the mathematics of computing parametric yield. Specifically, given n disturbance variables $\tilde{\zeta} = (\zeta_1, \ldots, \zeta_n) \in R^n$ which may affect the circuit performances, let $A \subset R^n$ be the set of disturbances $\tilde{\zeta}$, for which the circuit satisfies all specifications. The parametric yield, denoted as Y, is then

$$Y = \int_A f(\tilde{\zeta}) d\tilde{\zeta} = \int_{R^n} z(\tilde{\zeta}) f(\tilde{\zeta}) d(\tilde{\zeta}), \tag{1}$$

where the function $f(\tilde{\zeta})$ is the joint probability density function of disturbances, and $z(\tilde{\zeta}) = 1$ if $\tilde{\zeta} \in A$ and $z(\tilde{\zeta}) = 0$ otherwise.

Monte Carlo analysis can be used to estimate the parametric yield. In Monte Carlo analysis, a sample of disturbance variables,$\tilde{\zeta}$, is generated with the distribution,$f(\tilde{\zeta})$, and yield is estimated by averaging $z(\tilde{\zeta})$ for the sample. An alternative is to use surface integrals on the boundary of the acceptability region,A, as suggested in [3]. Both approaches require a very large number of expensive circuit simulations since $z(\tilde{\zeta})$ needs to be evaluated for each set of disturbances, $\tilde{\zeta}$, in the sample. Alternatively, statistical modeling methods have been proposed [4, 5] where each circuit performance is approximated by a computationally cheap surrogate response surface model. These methods only work well for low dimensional smooth circuit performances due to the fact that the number of simulations required for building the model and screening out the insignificant random variables to reduce dimensionality expands exponentially with the nonlinearity or order of the polynomial approximation and the number of significant variables. Recently, the GMDH algorithm [6] and nonparametric regression [7] have been proposed to model nonlinear functions. These approaches can be computationally intensive due to their lack of ability to sequentially sample and adaptively refine local regions of the model.

Using the fact that a highly nonlinear performance function can be approximated to arbitrary accuracy, *given a sufficiently fine partition of the domain defined by process*

[*] This work has been supported by NSF grant numbers MIP-9211407 and MIP-9501912.

33rd Design Automation Conference ®

DAC 96 - 06/96 Las Vegas, NV, USA ©1996 ACM, Inc. 0-89791-833-9/96/0006.. $3.50

disturbances and a sufficient number of simulations, we propose an approach to modeling performance functions which could be *highly nonlinear and have many variables* by efficiently and recursively partitioning the disturbance space into several regions, each of which is then modeled by a linear function. More specifically, we approximate the true performance function,$H(\tilde{\zeta})$,with a *piecewise linear* model

$$h(\tilde{\zeta}) = \sum_{k=1}^{m} h_k(\tilde{\zeta}) I(\tilde{\zeta} \in Q_k), \qquad (2)$$

where m is the number of disjoint hyper-rectangles, and $I(.)$ is a 0/1 valued function indicating if its argument is in hyper-rectangle Q_i. The union of these m Q_i's is R^n. h_k's are linear functions of $\tilde{\zeta}$, $h_k(\tilde{\zeta}) = a_0 + a_1\zeta_1 + \ldots + a_n\zeta_n$, where $a_i, i = 1 \ldots n$, are constants.

Linear models are much less sensitive to the dimension of the disturbance space compared to higher order polynomial models. As the dimension of the problem increases, we only need to increase the *number of regions* in the partition, rather than the number of terms in the model. This approach also allows us to model different regions of the model with different accuracies. This makes the yield estimation efficient, since for regions *less important* to yield estimation, fewer partitions and simulations are necessary and the models for these regions can be less accurate. In fact, the criteria we use for determining the importance of regions, combined with our partitioning approach, provides us with incremental knowledge of the performance function and the location of the boundary of A. As a result, we can select a minimum number of points for simulation to *adaptively* build the piecewise model. *The resulting model will be more accurate for the regions closer to the boundary of A and in regions close to nominal parameters.*

Another advantage of our approach is that it can exploit the low local dimensionality of the performance function; it can automatically screen out those less important disturbances locally during the building of the model.

2. Model Building

To obtain a piecewise linear approximation,$h(\tilde{\zeta})$, of the performance function (*e.g.* voltage gain), $H(\tilde{\zeta})$, we begin with N points, which are sampled using a Latin Hypercube scheme, $S = \{(\tilde{\zeta_1}, y_1), \ldots, (\tilde{\zeta_N}, y_N)\}$, where $y_j = H(\tilde{\zeta_j})$ and $\tilde{\zeta_j} = (\zeta_{1j}, \zeta_{2j}, \ldots, \zeta_{nj})$, for $j = 1, 2, \ldots, N$. Latin Hypercube sampling gives us an approximately uniform distribution of points in the input domain. We define our objective as fitting the sample data set S with the best piecewise linear surface. The best piecewise linear surface is one with a minimum overall prediction residual sum of squares ($PRESS$), which can predict the response at data points, $\tilde{\zeta}$,not in S.

2.1. The Recursive Partition and Local Linear Models

Formally, suppose at some stage of the model building process, the disturbance space is split into m regions, each of which has N_k data points in it, $k = 1, 2, \ldots, m$. The total $PRESS$ for this piecewise linear model is

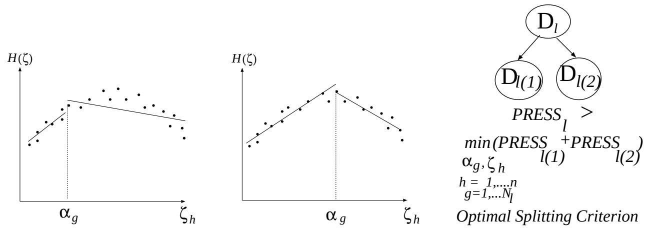
$$PRESS_t = \sum_{k=1}^{m} PRESS_k = \sum_{k=1}^{m}\sum_{j=1}^{N_k} (h_{k-\{j\}}(\tilde{\zeta_j}) - y_j)^2, \quad (3)$$

where $h_{k-\{j\}}(\tilde{\zeta_j})$ is the predicted function value at $\tilde{\zeta_j}$ with $\tilde{\zeta_j}$ removed from S.

$PRESS$ [8] essentially accumulates the importance of each point to the regression and hence is a good index for predictivity and a good criterion to find important variables (dimensions) whose domain needs to be split in building a piecewise-linear model. Such variables are locally significant to the nonlinearity of the underlying function because splitting their domain reduces the prediction error the most. Choosing any of these m regions to be split into two smaller regions might not always decrease $PRESS_t$. Therefore, $PRESS$ outperforms the residual sum of squares(RSS) as a splitting criteria. If we can not reduce $PRESS$ of a regional model, this means, based on the sampled data in this region, we cannot get a better predictive model. Then, accuracy can only be improved through resampling, or the estimate of yield for this region can be computed using traditional Monte Carlo methods. If resampling is performed, we may continue the process of splitting the domain to further reduce $PRESS_t$.

Achieving the optimal splitting of the space is important (Figure 1(a),(b)). Suppose the lth region is considered for splitting into two new regions $l(1)$ and $l(2)$ (Figure 1(c)). Suppose there are N_l data points, $D_l = \{(\tilde{\zeta_j}, y_j) | j = 1, 2, \ldots, N_l\}$,in the lth region, and $N_{l(1)}$ and $N_{l(2)}$ points in $D_{l(1)}$ and $D_{l(2)}$ respectively. The model for the lth region is denoted as $h_l(\tilde{\zeta}) = a_0 + \sum_{i=1}^{n} a_i\zeta_i$. $PRESS$ before splitting for this region is $\sum_{j=1}^{N_l}(h_{l-\{j\}}(\tilde{\zeta_j}) - y_j)^2$. We want to obtain two linear models, $f_{l(1)}$ and $f_{l(2)}$, forming a more accurate piecewise linear model for the lth region. If we constrain splitting for the hth variable,ζ_h(the hth component of $\tilde{\zeta}$), to coincide with a data point in the data set, $\zeta_{hj}, j = 1, 2, \ldots, N_l$, there are N_l ways of splitting the set D_l and n choices of variables. As a result, $N_l * n$ sets of two linear models for the lth region are possible. Specifically, for the hth variable,ζ_h, suppose that one of its N_l values,$\zeta_{hg} = \alpha_g, g \in 1, 2, \ldots, N_l$ is chosen to split the set D_l into $D_{l(1)}$ and $D_{l(2)}$. Then, the two linear models,$h_{l(1)} = b_0 + \sum_{i=1}^{N_{l(1)}} b_i\zeta_i$ and $h_{l(2)} = c_0 + \sum_{i=1}^{N_{l(2)}} c_i\zeta_i$ are used to fit $D_{l(1)} = \{(\tilde{\zeta_j}, y_j) | \tilde{\zeta_j} \in D_l, \zeta_{hj} \leq \alpha_g\}$ and $D_{l(2)} = \{(\tilde{\zeta_j}, y_j) | \tilde{\zeta_j} \in D_l, \zeta_{hj} > \alpha_g\}$, respectively. Each ζ_h and α_g are considered in turn and the best choice of the combination is made when $PRESS_l - PRESS_{l(1)} - PRESS_{l(2)}$ is maximal, where $PRESS_{l(k)} = \sum_{j=1}^{N_{l(k)}}(h_{l(k)-\{j\}}(\tilde{\zeta_j}) - y_j)^2$, k =1,2 (Figure 1(c)). If the same procedure is recursively applied to the l_1th and l_2th regions, we can gradually obtain an accurate piecewise linear model. Since linear regression is used, we can efficiently compute $PRESS$ without recomputing the model at $\tilde{\zeta_j}$, dropping y_j [4].

In summary, the procedure starts with building a global linear model, fit for all variables for the entire space. The domain is then recursively split. $PRESS$ is used to find the splitting which is optimal in the sense that the predictivity



(a) A Bad Split (b) A Good Split (c)

Figure 1. The Splitting Of The Space

is best improved and overfitting is prevented. The linear stepwise fit is applied to the leaf regions to improve the accuracy when the splitting process stops.

2.2. Criteria for Resampling and Checking the Overfitting

As we split the region into more pieces, each region will have less sampled points, therefore the ability to accurately prevent overfitting,*i.e.* too close fitting of the model resulting in degraded estimates with poor predictive performance, is reduced even with $PRESS$ as the splitting criteria. As a remedy, we propose using a checking data set. Suppose $\tilde{h}_c = h(\tilde{\zeta})$ and $\tilde{H}_c = H(\tilde{\zeta})$ correspond to the checking data set and are the predicted and true response vectors, respectively. The correlation coefficient,ρ_{check}, as defined in [7], and $R^2_{press} > 0.85$ serve as two criteria for measuring the accuracy of the model.

Essentially, we use splitting and resampling of regions to improve the model accuracy sequentially. For efficiency, resampling a region is only *considered* when the model is not accurate enough and the number of points is not sufficient for splitting of the domain, and is only *done* if statistical modeling is likely to be more efficient than Monte Carlo methods for computing yield for a given region. Specifically, the number of points used to estimate yield with the Monte Carlo method is computed for a given region. If this number is smaller than the number of simulations required for splitting and modeling,*i.e.* twice the number of variables plus the number of sampled data currently in the region, it is not worthwhile to build a statistical model for the region. The algorithm is outlined in Figure 2.

3. Yield Estimation

3.1. Adaptively Refining the Model

If a circuit is required to satisfy q specifications, in the presence of imperfect models, the yield can be estimated by [5]:

$$\hat{Y} = \frac{1}{N_{mc}} \sum_{j=1}^{N_{mc}} E(z(\tilde{\zeta_j})) = \frac{1}{N_{mc}} \sum_{j=1}^{N_{mc}} \prod_{t=1}^{q} P_t(\tilde{\zeta_j}) \qquad (4)$$

where $P_t(\tilde{\zeta_j})$ is the probability that a specification,t, is satisfied at $\tilde{\zeta_j}$. The average variance associated with the computation of $E(z(\tilde{\zeta_j}))$ for all N_{mc} points in all regions is

$$\frac{1}{N_{mc}} \sum_{j=1}^{N_{mc}} Var(z(\tilde{\zeta_j})) = \frac{1}{N_{mc}} \sum_{j=1}^{N_{mc}} E(z(\tilde{\zeta_j}))(1 - E(z(\tilde{\zeta_j}))). \quad (5)$$

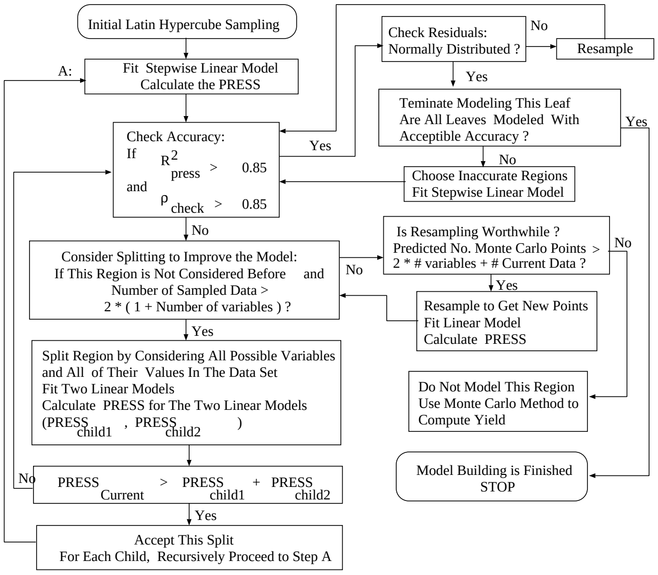

Initial Latin Hypercube Sampling
A:
Fit Stepwise Linear Model
Calculate the PRESS
Check Accuracy:
If R^2_{press} > 0.85
and ρ_{check} > 0.85
Yes
Check Residuals:
Normally Distributed ?
No
Resample
Yes
Teminate Modeling This Leaf
Are All Leaves Modeled With
Acceptible Accuracy ?
Yes
No
Choose Inaccurate Regions
Fit Stepwise Linear Model
No
Consider Splitting to Improve the Model:
If This Region is Not Considered Before and
Number of Sampled Data >
2 * (1 + Number of variables) ?
No
Is Resampling Worthwhile ?
Predicted No. Monte Carlo Points >
2 * # variables + # Current Data ?
No
Yes
Resample to Get New Points
Fit Linear Model
Calculate PRESS
Yes
Split Region by Considering All Possible Variables
and All of Their Values In The Data Set
Fit Two Linear Models
Calculate PRESS for The Two Linear Models
($PRESS_{child1}$, $PRESS_{child2}$)
Do Not Model This Region
Use Monte Carlo Method to
Compute Yield
No
$PRESS_{Current}$ > $PRESS_{child1}$ + $PRESS_{child2}$
Model Building is Finished
STOP
Yes
Accept This Split
For Each Child, Recursively Proceed to Step A

Figure 2. Flow Chart For Model Building

If the piecewise linear model is accurate, this quantity should be small. If this quantity is too large, then the model for at least one specification needs to be improved over some critical regions. This inaccurate specification should have many points in the Monte Carlo sample that are close to the boundary of A. Points that are close to the boundary will have a larger than average contribution to (5). Let us call this set of boundary points G,

$$G = \{\tilde{\zeta_j} | Var(z(\tilde{\zeta_j})) > \frac{1}{N_{mc}} \sum_{j=1}^{N_{mc}} Var(z(\tilde{\zeta_j}))\}. \qquad (6)$$

From this subset of the Monte Carlo sample set, the specification that needs improvement most maximizes

$$Var(Spec_t) = \sum_{\tilde{\zeta_j} \in G} P_t(\tilde{\zeta_j})(1 - P_t(\tilde{\zeta_j})). \qquad (7)$$

If the specification, t, is most critical,*i.e.* maximizing (7), we improve the model in the most critical region, *i.e.* region with the largest contribution to (7). Specifically, suppose that for the specification, t, the space has been partitioned into m regions and the lth region is the most critical region, then this region maximizes

$$Var(R_l) = \sum_{\tilde{\zeta_j} \in G \cap R_l} P_t(\tilde{\zeta_j})(1 - P_t(\tilde{\zeta_j})), \qquad (8)$$

where $\tilde{\zeta_j} \in R_l$. This quantity would be large if *either there are many Monte Carlo sampling points in the region or many points in this region are close to the boundary of A*. Therefore $Var(R_l)$ measures the importance of a region and identifies regions where the model needs to be refined.

3.2. The Yield Estimation Algorithm

The algorithm is outlined in Figure 3. In the first stage, rough piecewise linear approximations are made of all performance functions over the entire disturbance space by the model building method described in section 2. R^2_{press} and ρ_{check} are used to measure the goodness-of-fit. Then, yield (4) and average yield variance due to statistical modeling (5) are estimated. If the variance is higher than desired, the model is refined efficiently, *i.e.* only the most critical functions, identified using (7), are approximated more accurately in the most critical regions, determined using (8). This process continues until average modeling variance (5) is sufficiently small. In this adaptive way, the algorithm builds a piecewise linear model which approximates the true function to an arbitrary desired accuracy.

4. Computational Complexity

The computations in building the model involve computations in performing linear regression, calculating $PRESS$, ρ_{check}, R^2, and in performing stepwise linear regression. These computations are applied to each region and are recursively repeated for each region of the domain splitting until the model is accurate enough. Linear regressions are performed to compute $PRESS$ and then find the optimal split. Once the optimal split is found, we perform stepwise linear regression for the two new regions after the split to get further improvement of the accuracy. Let us begin by analyzing the complexity involved in a single splitting. Suppose N is the size of the data set and n is the dimension. Setting up the normal equation for the linear regression involves $O(Nn^2)$ multiplications. Solving requires $O(n^3)$ multiplications, and evaluating the $PRESS$ of the model needs $O(Nn^2)$ operations. Note that the matrix $X(X^{'}X)^{-1}X^{'}$ already obtained from fitting the linear model can be used to compute $PRESS$ using the method mentioned in section 2. Evaluating R^2 and ρ_{check} requires $O(Nn)$ multiplications. Finding a best split involves considering Nn splits, hence the cost of finding the best split and fitting two models is $O(N^2n^3 + Nn^4)$. The number of best splits in building the model is one less than the number of regions, specified as m, in the final model. Since the complexity of fitting a stepwise linear regression equation is $O(n^2N + n^3)$ [9], the total cost of performing stepwise linear regressions is $O(mn^2N+mn^3)$. Note that m can not be known in advance because it depends on the nonlinearity of the function and the accuracy of the model to be achieved. Therefore, the overall cost for building the model is $O(m(N^2n^3 + Nn^4))$. Overall N might be large. Nevertheless, computations are only based on local small subsets of the whole data set since recursive splitting of the space is involved. Hence, this computed complexity is an upper bound.

Compared with the polynomial regression approach, the number of final regions in the piecewise linear model increases much slower than number of terms used in polynomial regression, because for an n dimension problem, a pth order polynomial requires $(n + 1)(n + 2)...(n + p)/p!$ terms. This makes our approach use much fewer circuit simulations. Furthermore, in estimating the yield, some regions need not be modeled very accurately, and this makes the number of final regions even smaller since some unimportant regions can be left unmodeled. Besides, we avoid other problems with polynomial regression such as the observations that the design matrix tends to be ill-conditioned as the order of polynomial increases, and that higher-order polynomials tend to loose predictivity ability. In general, the use of high-order polynomials (> 2) should be avoided [10].

5. Experimental Results

Our algorithm has been applied to model a a variety of high dimensional and highly nonlinear functions using a SUN SPARC II IPX machine.

5.1. Example 1

This is a 10-dimensional function used in [7] with 5 noise variables independent of the response, a high-order interaction between x_0 and x_1, and a nonlinear function of x_2,

$$\begin{aligned} y_i = 0.02e^{4x_0+3x_1} + 5sin(\frac{\pi}{2}x_2) + 3x_3 + 2x_4 + 0x_5 \\ +0x_6 + 0x_7 + 0x_8 + 0x_9 \end{aligned}$$

where $-1.5 \leq x_i \leq 1.5$.

To observe the adaptability of our approach for estimating the yield, we assume this function models some real performance function of a circuit, and the *specification* is $y_i \leq 5$. The initial global linear fit was very inaccurate with R^2=0.49 andR^2_{press}=0.21, which shows that the function is highly nonlinear for this range of the x_i's. From Table 1, we can see that the model improves as splitting and resampling are performed. The variables chosen for splitting are either x_0 or x_1, which means our approach indeed identifies the most significant nonlinearity relating to these two variables. In total, we used 160 data points to obtain the final model which has 6 leaves, four of which are critical regions with R^2 above 0.95. The time needed was 40 seconds, which is much less than 100 seconds on the same machine in [7]. Our yield estimation algorithm, based on 2000 Monte Carlo points, gives us a yield of 98%. The Crude Monte Carlo estimate gives us the yield of 96%.

5.2. Example 2

We use the same example as in [7] to check how well our approach handles functions with abrupt changes, which could occur in circuit performances as process variations deviate too far from normal values and devices change their regions of operations. Abrupt functions are highly nonlinear and not additive, therefore, recent nonlinear nonparametric approaches [7, 11] based on additive models are unsatisfactory. The function is $z = f(x, y)$

$$\begin{aligned} z = 1 \qquad if x^2 + y^2 \leq 0.5 \\ z = 0 \qquad if x^2 + y^2 > 0.5 \end{aligned}$$

where $-1.0 \leq x_i \leq 1.0$ as in [7].

Both R^2_{press} and R^2 of the initial global linear model are 0, therefore 30 additional data were used as checking data initially. Using splitting and resampling, this approach gives us a final piecewise-linear model with 13 leaf regions, each of which is perfectly fit with the sampled data. Figure 4 shows the true function, an intermediate model with 7 regions, and the final model with 13 regions. The model is accurate and captures the sharpness very well due to the partitioning ability of our approach. Compared with the result from [7], it can be seen that a regression spline approach produces error in almost all regions and large errors near the sharp changes, while our approach adaptively refines the model near the sharp regions. The resampling and checking data set are generated locally several times during modeling. The number of data used is 195, which is very close to 200 in [7].

5.3. Example 3

This next example is a third-order Butterworth analog low-pass filter (Figure 5). This filter contains one two-stage CMOS operational amplifier with a cascaded first stage and several resisters and capacitors outside the amplifier. The specifications are:

Frequency at gain=-3dB > 900 Hz
Power Supply Rejection Ratio (PSRR) @ 1kHz > 72dB

These two performances vary nonlinearly due to manufacturing process fluctuations which, for this circuit, were characterized by 45 independent normally distributed disturbance variables, including 17 variables modeling interdie variations and 28 variables modeling mismatch between transistors. Interdie and intradie variations were described by variations in channel length (L), width (W), zero-bias threshold voltage (V_{to}), oxide thickness (T_{ox}), doping density of substrates (N_{sub}) , lateral diffusion length (ld), the surface mobility (u_o), resistors (R_1 , R_2, R_3), and capacitors (C_1, C_2, C_3).

Given this characterization and the desired average modeling variance of at most 0.0012, our algorithm automatically modeled the performances and computed the parametric yield. Initially, 148 data points were generated, simulated, and poorly fit by a linear model with R^2=0.86 and R^2_{press}=0.72. Therefore, 70 additional data points were generated and simulated as the checking data set. For the corner frequency, after the first split, we found a highly nonlinear region and a weakly linear region. This two piece linear model was accurate enough to estimate the variance of the predicted response. Since the highly nonlinear region had only a few points for Monte Carlo yield estimation, it was not worthwhile to refine the model and instead circuit simulation was used for yield estimation.

The performance, PSRR, was found to be nonlinear and more critical than the corner frequency specification for yield estimation. Therefore it was refined using another 180 simulations. The final model has 4 leaves in which R^2, R^2_{press}, and ρ_{check} are all above 0.98. A comparison of results from different approaches to estimating yield is shown in Table 2. Note that, for such a high dimension, at least 1081 simulations are required to model just one performance function using a quadratic polynomial model. Consequently, a comparison is only made with linear modeling and crude Monte Carlo analysis.

6. Conclusions

In this paper, we presented an algorithm that can efficiently compute the yield and its variance for analog circuits relying on an adaptive approach to modeling in which the model can be improved locally as more data is obtained sequentially in critical regions and no functional form of the model needs to be assumed *a priori*. Partitioning the space, simple first-order model fitting, and traversing the binary tree are very fast, easy to automate, and consequently introduce trivial computational overhead over the cost of circuit simulations. Consequently, the approach provides an efficient way of estimating yield for high dimensional problems. Examples indicate a significant speed-up over standard Monte Carlo methods and the models of performance functions are more accurate than is possible with regression.

References

[1] D. Eshbaugh, "Generation of correlated parameters for statistical circuit simulation," *IEEE Trans. Computer-Aided Design*, vol. CAD-11, pp. 1198–1206, Oct. 1992.

[2] C. Michael and M. Ismail, "Statistical modeling of device mismatch for analog MOS integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 154–165, Jan. 1992.

[3] P. Feldmann and S. W. Director, "Improved methods for ic yield and quality optimization using surface integrals," *Proc. ICCAD*, pp. 158–161, Nov. 1991.

[4] K. K. Low and S. W. Director, "An efficient methodology for building macromodels of IC fabrication processes," *IEEE Trans. Computer-Aided Design*, vol. CAD-8, pp. 1299–1313, Dec. 1989.

[5] L. Milor and A. Sangiovanni-Vincentelli, "Computing parametric yield accurately and efficiently," *Proc. ICCAD*, pp. 116–119, 1990.

[6] A. Strojwas and S. Director, "An efficient algorithm for parametric fault simulation of monolithic IC's," *IEEE Trans. Computer-Aided Design*, vol. CAD-5, pp. 5–14, Jan. 1986.

[7] C. Y. Chao and L. Milor, "Performance modeling of analog circuits using additive regression splines," *IEEE Trans. Semiconductor Manufacturing*, Aug. 1995.

[8] D. M. Allen, "Mean square error of prediction as a criterion for screening variables," *Technometrics*, vol. 13, pp. 469–475, 1971.

[9] A. Ralston, *Mathematical methods for digital computers.* Wiley, NY, chapter 17 ed., 1960.

[10] D. C. Montgomery and E. A. Peck, *Introduction to linear regression analysis.* Wiley, NY, pp. 183 ed., 1982.

[11] J. Friedman, "Multivariate additive regression splines," *The Annals of Statistics*, vol. 19, pp. 1–141, 1991.

```
Stage 1
INPUT    V ( the desired variance of the yield estimate )
INPUT    N ( the initial number of points to be generated and simulated )
For each specification {
        While ( $R_{press}$ < 0.85 and $\rho_{check}$ < 0.85 ) {
                Build the piecewise-linear model
        }
}
Compute the parametric yield (4) and its variance (5)
A few real circuit simulations are invoked by the Monte Carlo algorithm
only for regions not worth modeling

Stage 2
While ( yield variance > V ) {
        Determine the specification with the largest variance using (7)
        For this specification {
                Determine the critical regions in disturbance space using (8)
                Latin Hypercube points are generated in these critical regions
                Resume the model building process to refine the models in
                these regions
        }
}
OUTPUT   parametric yield and yield variance
```

Figure 3. Yield Estimation Algorithm

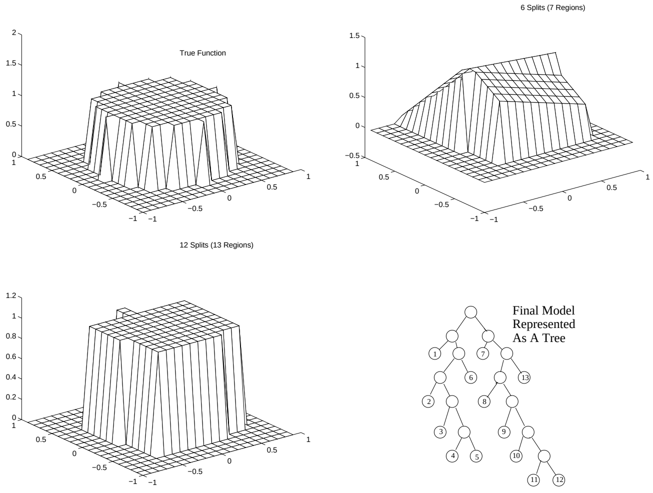


Figure 4. The Modeling Process For Example 2

Method	Estimated yield	Time	No. of simulations used
Monte Carlo (Crude)	56.5%	334 min	4000
Linear Model	69.7%	Modeling: 12 sec Simulation: 24 min	296
Piecewise Linear	57.2%	Modeling: 15.8 min Simulation: 58.5 min	706

Table 2. Yield Estimation in Example 3

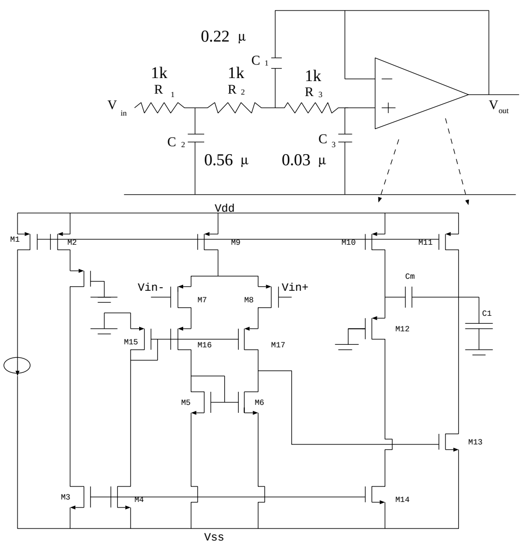


Figure 5. The Butterworth Low-Pass Filter

<table>
<tr><td colspan="6">Initial global linear model:
$Press$=3264.8, R^2=0.4878, R^2_{press}=0.2143</td></tr>
<tr><td colspan="6">Piece-wise linear model:
6-region model resulting from alternating splits of x_0 and x_1</td></tr>
<tr><td>$Leaf$ $no.$</td><td>$Press$</td><td>R^2</td><td>R^2_{press}</td><td>$\rho^\dagger$</td><td>$MC^\ddagger$</td></tr>
<tr><td>1</td><td>72.0316</td><td>0.9550</td><td>0.9127</td><td>0.9539</td><td>1607</td></tr>
<tr><td>2</td><td>87.9873</td><td>0.9568</td><td>0.8067</td><td>0.9392</td><td>266</td></tr>
<tr><td>3</td><td>71.6457</td><td>0.9712</td><td>0.8966</td><td>0.9490</td><td>83</td></tr>
<tr><td>4</td><td>19.5349</td><td>0.9712</td><td>0.9478</td><td>0.9423</td><td>21</td></tr>
<tr><td>5</td><td>19.8087</td><td>0.9951</td><td>0.9878</td><td>0.4021</td><td>2</td></tr>
<tr><td>6</td><td>0.0205</td><td>1.0</td><td>1.0</td><td>-0.217</td><td>21</td></tr>
<tr><td colspan="6">Region 1 : x_0 &lt; 0.3561
Region 2 : 0.3561 $\le x_0$ &lt; 0.9457, x_1 &lt; 0.1957
Region 3 : 0.3561 $\le x_0$ &lt; 0.7048,0.1957 $\le x_1$ &lt; 1.0376
Region 4 : 0.7048 $\le x_0$ &lt; 0.9457,0.1957 $\le x_1$ &lt; 1.0376
Region 5 : 0.3561 $\le x_0$ &lt; 0.9457,1.0376 $\le x_1$
Region 6 : 0.9457 $\le x_0$</td></tr>
<tr><td colspan="6">Overall modeling time : 40 sec
Data used in modeling: 160
Data used in yield estimation: 183
†: Corr. coef. for 2800 Latin Hypercube points
‡: No. Monte Carlo data points (out of 2000)</td></tr>
</table>

Table 1. Summary of the Modeling Result for Example 1