Globally Reliable Variation-Aware Sizing of Analog Integrated Circuits via Response Surfaces and Structural Homotopy

Trent McConaghy, *Member, IEEE*, and Georges G. E. Gielen, *Fellow, IEEE*

***Abstract*—This paper presents SANGRIA, a tool for automated globally reliable variation-aware sizing of analog integrated circuits. Its keys to efficient search are adaptive response surface modeling, and a new concept, *structural homotopy*. Structural homotopy embeds homotopy-style objective function tightening into the search state's structure, not dynamics. Searches at several different levels are conducted simultaneously: The loosest level does nominal dc simulation, and tighter levels add more analyses and {process, environmental} corners. New randomly generated designs are continually fed into the lowest (cheapest) level, always trying new regions to avoid premature convergence. For further efficiency, SANGRIA adaptively constructs response surface models, from which new candidate designs are optimally chosen according to both yield optimality on model and model prediction uncertainty. The *stochastic gradient boosting* models support arbitrary nonlinearities, and have linear scaling with input dimension and sample size. SANGRIA uses SPICE in the loop, supports accurate/complex statistical SPICE models, and does not make assumptions about the convexity or differentiability of the objective function. SANGRIA is demonstrated on four different analog circuits having from 10 to 50 devices and up to 444 design/process/environmental variables.**

***Index Terms*—Analog, design automation, integrated circuit, process variation.**

I. Introduction

UNCONTROLLABLE factors in semiconductor manufacturing—process variations—have always existed. Up until recently, the effects would cancel out across the billions or more atoms in a given transistor. However, transistors have shrunk so much that even a single atom out of place can affect a transistor's behavior, leading to worsened circuit behavior and even circuit failure. The variation is already large, and will continue to get worse with future process technologies [1]. Such variation is particularly problematic for analog circuits, which do not have the abstraction of binary digits to hide small variations. Process variations are not the only problem. Layout parasitics, aging/reliability, electromagnetic compatibility, proximity, and other phenomena can affect circuit behavior. However, because of their direct short-term impact on circuit yields, addressing process variations is the most urgent.

Design of robustly behaving analog circuits is difficult and time consuming. This has caused the analog portion of chips to become the design bottleneck [1]. Yet, we cannot ignore or bypass analog circuits, since they are crucial for digital circuits to interface with the real world. As of 2006, 70% of systems on chips or systems in packages have some analog functionality, up from 50% in 2005 and 10% in 1999 [2]. We need a means to design analog circuits which meet performance goals, have high yield, with low area, all designed fast enough to succeed in tight time-to-market schedules [3].

One option for designing robust analog circuits is to simply design with worst-case corners. However, this can give unacceptable performance margins or area increase, even if best-practice layout techniques are used. Automated variation-aware sizing is a promising alternative, and there has accordingly been much recent research on the topic [4]–[12].

An industrially useful optimization tool must possess several characteristics. 1) It must be SPICE accurate, support accurate/complex statistical models such as [13], and reflect these within its objective function. 2) It should be globally reliable—the user should not need to worry about whether the algorithm is stuck at a local optimum (and there is a difference between a *nominal* optimum and a *statistical* optimum, as Section II-B discusses). 3) Because the true objective function mapping is not known, the tool should not make assumptions about the convexity or continuity of the mapping. 4) Finally, it should be able to scale to handle dozens of devices, dozens of design variables, and hundreds of process variables.[1] Using a cluster for parallel computing is acceptable.

As Table I summarizes, none of the existing approaches to yield optimization possesses all of these characteristics (Section III has details). A new approach is needed, which is the focus of this paper. The novel contributions of this paper are the following.

1) An analog yield optimization approach that, unlike other approaches, has the characteristics of: a) accurate variation model; b) escapes local yield/Cpk optima; c) handles nonconvex/discontinuous mappings; and d) scales well.

Manuscript received December 8, 2008; revised March 16, 2009, May 22, 2009, and August 6, 2009. Current version published October 21, 2009. This work was supported in part by IWT/Medea+ Uppermost, by Solido Design Automation, Inc., and by FWO Flanders. This paper was recommended by Associate Editor P. Li.

T. McConaghy was with the Department of Electrotechnical Engineering—Microelectronics and Sensors (ESAT—MICAS), Katholieke Universiteit Leuven, 3001 Leuven, Belgium. He is now with Solido Design Automation, Inc., Saskatoon, SK S7N 3R3, Canada (e-mail: trent_mcconaghy@yahoo.com).

G. G. E. Gielen is with the Department of Electrotechnical Engineering—Microelectronics and Sensors (ESAT—MICAS), Katholieke Universiteit Leuven, 3001 Leuven, Belgium.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2009.2030351

[1]These are numbers for cell-level design. System-level design can be handled through an appropriate hierarchical design methodology; that is beyond the scope of this paper.

TABLE I
COMPARISON OF ANALOG YIELD OPTIMIZATION APPROACHES

Approach	Accurate model of variation within objective function	Escapes local *yield / cpk* optima	Handles nonconvex and discontinuous func. mappings	Will scale to dozens of design vars. and hundreds of process vars.
FF/SS corners	N	N	Y	Y
Semi-infinite programming corners [4]	N	N	Y	N
Device operating constraints [5], [6]	N	N	Y	Y
Convex polytope design centering / WCD [7], [8]	≈Y	N	N	Y?
Nominal tradeoffs first [9], [10]	Y	N	Y	Y?
Proj.-based model-building optimization [11]	Y	N	N	Y?
Kriging-based model-building optimization [12]	Y	Y	Y	N
SANGRIA (this paper)	**Y**	**Y**	**Y**	**Y**

2) An enabling aspect of the algorithm is a novel homotopy [14] approach called *structural homotopy* which continually explores new regions of design space with loose evaluation, refines promising designs with successively more evaluation, and fully evaluates the most promising designs. It searches these different levels in the exploration–exploitation spectrum *simultaneously*. Since fresh regions are continually explored and refined, the algorithm does not prematurely converge to local optima.
3) A second enabling aspect of the algorithm is model-building optimization (MBO) with several novel aspects. It uses *stochastic gradient boosting* (SGB) [15] for models. SGB has linear scaling with input dimension and sample size [15], yet can handle nonconvex and discontinuous mappings. To our knowledge, this is the first time that SGB has been used for MBO, in analog sizing, or otherwise. Unlike other MBO approaches in the literature, model uncertainty is computed with *ensembles* of SGB models, and employed within a Pareto-aware *multiobjective* optimization [16] to find candidate designs that tradeoff optimality (on the model) with model uncertainty.

The approach is called SANGRIA: **S**tatistical, **a**ccurate, a**n**d **g**lobally **r**eliable s**i**zing **a**lgorithm.

The rest of this paper is organized as follows. Section II describes the yield optimization problem. Section III reviews past approaches to yield optimization. Sections IV and V describe homotopy and response-surface-based optimization, respectively. Section VI describes the SANGRIA algorithm. Section VII gives experimental results for SANGRIA on a suite of optimization problems, on circuits having hundreds of variables. Section VIII concludes this paper.

II. YIELD OPTIMIZATION PROBLEM

A. Problem Formulation

Given a design space D, process parameter space S with distribution $pdf(\boldsymbol{s})$, environmental space Θ, and measurable performances with associated specifications $\boldsymbol{\lambda}$, the aim is to find a vector-valued design point $\boldsymbol{d}^*$ that maximizes the objective f

$$\boldsymbol{d}^* = \arg\max_{\boldsymbol{d}\in D} \{f(\boldsymbol{d})\} \tag{1}$$

where the design space $D = \otimes_{i=1}^{N_d}\{D_i\}$ having continuous or discrete variables that include transistor widths W, transistor lengths L, resistances R, etc. The range for each variable is determined by technology process constraints and the user's setup. The objective f is a statistical robustness estimator. It can be yield Y, which is the expected proportion E of feasible circuits δ across the distribution of manufacturing variations $pdf(\boldsymbol{s})$

$$Y(\boldsymbol{d}) = E\{\delta(\boldsymbol{d},\boldsymbol{s})|pdf(\boldsymbol{s})\} = \int_{\boldsymbol{s}\in S} \prod_{i=1}^{N_g} \delta_i(\boldsymbol{d},\boldsymbol{s}) * pdf(\boldsymbol{s})\, d\boldsymbol{s} \tag{2}$$

where the possible manufacturing variations $S = \Re^{N_s}$ include variations in oxide thickness $t_{\rm ox}$, substrate doping concentration $N_{\rm sub}$, etc. These can be on a per-device level (local), or across the circuit or wafer (global). For an accurate model, both local and global variations must be modeled. $\boldsymbol{s}$ describes the variations in a single manufactured design, i.e., "process corner." δ_i is the feasibility of instance $\{\boldsymbol{d},\boldsymbol{s}\}$ at constraint i

$$\delta_i(\boldsymbol{d},\boldsymbol{s}) = I\left(g_{{\rm wc},i}(\boldsymbol{d},\boldsymbol{s}) \le 0\right) \tag{3}$$

where $I(condition)$ returns one if *condition* is True (feasible), and zero otherwise (infeasible). The quantity $g_{{\rm wc},i}$ is the worst-case constraint value across possible environmental conditions Θ

$$g_{{\rm wc},i}(\boldsymbol{d},\boldsymbol{s}) = \min_{\theta\in\Theta}\{g_i(\boldsymbol{d},\boldsymbol{s},\boldsymbol{\theta})\} \tag{4}$$

where $\Theta = \{\Re^{N_\theta} | \theta_{j,\min} \le \theta_j \le \theta_{j,\max}; j = 1,\ldots,N_\theta\}$. Environmental variables include temperature T, power supply voltage $V_{\rm dd}$, and load resistance $R_{\rm load}$. $\boldsymbol{\theta}$ is an "environmental corner." Each constraint g_i corresponds to a performance specification λ_i which has an aim and a threshold, and translates into an inequality constraint. For example, $\lambda_1 = \{power \le (1e-3)W\} \mapsto \{g_1 \le 0; g_1 = power - (1e-3)\}$.

Performances can be measured by SPICE circuit simulation, equations, or other means. A testbench ξ specifies how to extract one or more performance measures at a given circuit design, process point, and environmental point. All testbenches are $\boldsymbol{\xi} = \{\xi_1,\xi_2,\ldots,\xi_j,\ldots,\xi_{N_\xi}\}$, to measure all performances $\boldsymbol{\lambda}$. The environmental space is actually testbench-dependent: $\Theta_j = F(\xi_j)$. For example, some testbenches may have loads that other testbenches do not have. Each testbench ξ_j has a representative set of environmental corners $\widehat{\Theta_j} \approx \Theta(\xi_j)$ where $\widehat{\Theta_j} = \{\boldsymbol{\theta_{j,k}}\}, k = 1,\ldots,N_c(j)$.

"Process capability" (Cpk) [17] is an alternative to yield for the objective f. Cpk simultaneously captures the worst performance's spread and the margin above/below its specification. Therefore, unlike yield, Cpk can distinguish between two designs having a yield of 0%, or between two designs having (estimated) yield of 100%. Cpk is defined as the worst-case Cp_i across all constraints

$$Cpk(\boldsymbol{d}) = \min_{i \in 1,2,\ldots,N_g} \{Cp_i(\boldsymbol{d})\} \tag{5}$$

where Cp_i is

$$Cp_i(\boldsymbol{d}) = \left(E\left(g_{i,\mathrm{wc}}(\boldsymbol{d})\right) - 0\right) / \left(3 * \sigma\left(g_{i,\mathrm{wc}}(\boldsymbol{d})\right)\right) \tag{6}$$

where E is the expected value of $g_{i,\mathrm{wc}}$ across $\boldsymbol{s}$, and σ is the corresponding standard deviation.

B. Local Versus Global Optimization

This section highlights the importance of global *statistical* optimization, i.e., optimizing without premature convergence at local yield/Cpk optima. The user wants the optimizer to keep running until a target is hit (e.g., target yield), or to get the best possible design subject to computational/time resources (including the best design if the target is not achievable). While the algorithm is running, the user should be able to trust its convergence—the user should not need to worry about whether the algorithm is stuck at a local optimum.

A popular way to "be global" is to first do *global* optimization on the nominal objective(s), followed by local yield/Cpk optimization. This will not always work, as we now illustrate. Fig. 1 shows a simple yield optimization problem setup, where the nominal performance is a multimodal function of $W1$ (top half). Process variation is modeled by simply adding a Gaussian-distributed random variable to $W1$, leading to a mapping of $W1$ to yield, as shown in Fig. 1 (bottom). A nominal optimizer would return a $W1$ corresponding to the tall narrow hill of Fig. 1 (top); then starting there, the local yield optimizer will return a similar $W1$ value for the top of the short hill of Fig. 1 (bottom), i.e., a *global nominal* optimum led to a *local* yield optimum, i.e., *yield optimization is stuck at a local optimum*, which is undesirable. The problem can actually be far worse, when design and process variables do not have such a simple additive relationship.

We want an algorithm that the user can trust to continually converge without getting stuck at local optima. To achieve this, the search algorithm must consider global search and yield search *simultaneously*—it cannot separate nominal and statistical.

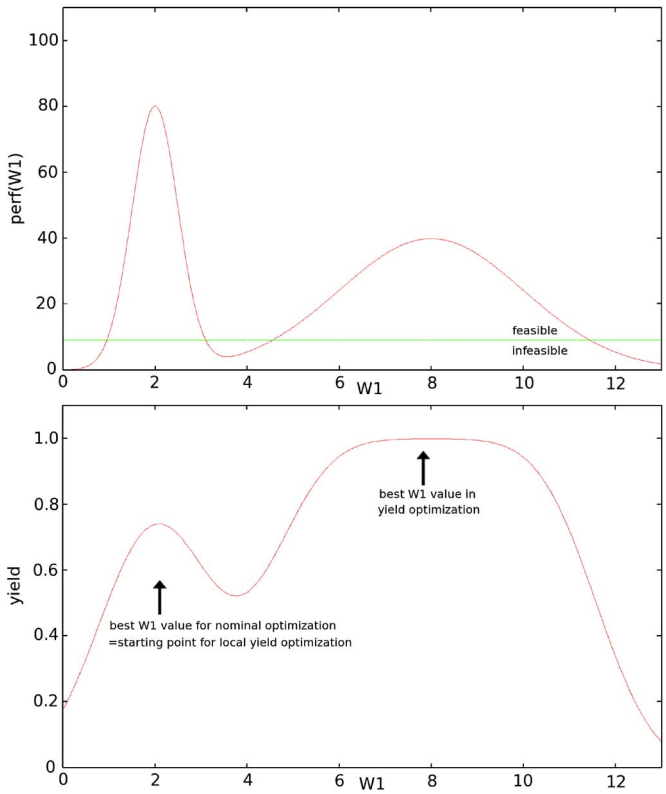

Fig. 1. Multimodality in performance space can lead to multimodality in yield space or disjoint feasible regions. In this conceptual example, the global nominal optimum will lead to a local optimum for yield.

III. Review of Yield Optimization Approaches

A. Yield Optimization Using Direct MC

This can be considered a "baseline" approach. An optimization algorithm explores the design space D

$$\boldsymbol{d}^* = \arg\max_{d \in D} \left(\widehat{Y_{\mathrm{MC,sim}}}(\boldsymbol{d})\right) \tag{7}$$

where $\widehat{Y_{\mathrm{MC,sim}}}$ is estimated with Monte Carlo (MC) sampling and simulation. In MC sampling, N_{MC} process points i are drawn from the process distribution $\boldsymbol{s_i} \sim pdf(\boldsymbol{s})$. SPICE simulation is done at design point $\boldsymbol{d}$, for each process point $\boldsymbol{s_i}$, for each testbench ξ_j, for each environmental point $\boldsymbol{\theta_{j,k}}$, giving performance vectors $\boldsymbol{\lambda_{i,j,k}}$ and corresponding constraint-value vectors $\boldsymbol{g_{i,j,k}}$. From the simulation data, $\widehat{Y_{\mathrm{MC,sim}}}$ is the average estimated feasibility across samples

$$\widehat{Y_{\mathrm{MC,sim}}}(\boldsymbol{d}) = \frac{1}{N_{\mathrm{MC}}} * \sum_{i=1}^{N_{\mathrm{MC}}} \widehat{\delta}_i(\boldsymbol{d}, \boldsymbol{s_i}) \tag{8}$$

where $\widehat{\delta_i} = \widehat{\delta}(\boldsymbol{d}, \boldsymbol{s_i})$ is the feasibility of sample $\boldsymbol{s_i}$. $\widehat{\delta_i}$ has value one only if at each testbench j, all constraints l at all environmental corners k are feasible

$$\widehat{\delta}(\boldsymbol{d}, \boldsymbol{s_i}) = \prod_{j=1}^{N_\xi} \left\{ \prod_{l=1}^{N_g(j)} I\left(\min_k \{g_{i,j,k,l}\} \le 0\right) \right\}. \tag{9}$$

We examine the typical runtime. If the time to simulate the most expensive testbench (e.g., tran) is 1 min, $N_c = 8$ environmental corners per testbench, $N_{\mathrm{MC}} = 50$ process points, and five simulators in parallel, then the total simulation time to evaluate one design $= (1\text{ min}) * 8 * 50/5 = 80$ min. If an optimization algorithm explores 1000 designs, then direct MC optimization will take $80 * 1000 = 80\,000$ min $= 55$ days.

The advantage of direct MC on SPICE is simplicity and accuracy of results, but it has the major disadvantage of runtime.

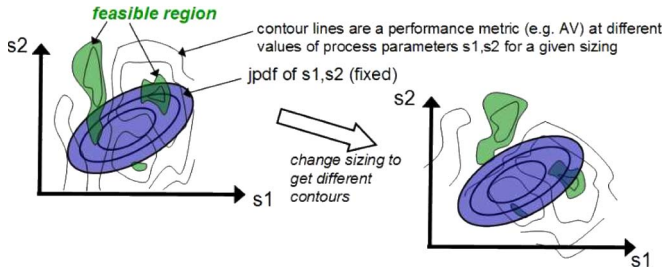

Fig. 2. Design-centering view of yield optimization.

Using symbolic [18] or regression models [19] as a substitute for the SPICE simulations improves runtime a bit, but can be inaccurate and is difficult to do for many input parameters [19], [20]. Adaptive-modeling approaches such as [12] and [21] choose new samples by maximizing the predicted performances of the circuit. However, this can get stuck in a local optimum due to the model's blind spots [22].

B. Yield Optimization Using Corners

The core idea of all corners-based approaches is as follows: If corners are "representative" of process and environmental variations, and all corners can be "solved," then the final design's yield will be near 100%

$$\boldsymbol{d}^* = \arg\max_{d \in D} \left(\prod_{\Xi_i \in \boldsymbol{\Xi}} \delta(\boldsymbol{d}, \Xi_i) \right) \mapsto Y(\boldsymbol{d}^*) \approx 100\% \quad (10)$$

where to "solve at a corner" means to find a design $\boldsymbol{d} \in D$ which is feasible across all constraints at the corner Ξ_i, i.e., $\delta(\boldsymbol{d}, \Xi_i) = 1$. To "solve at corners" means to find a design that is feasible at all those corners $\{\delta(\boldsymbol{d}, \Xi_1) = 1, \delta(\boldsymbol{d}, \Xi_2) = 1, \ldots\}$. Different approaches have different choices for "representative" corners, but they are either inaccurate (e.g., FF/SS) or too pessimistic (e.g., semiinfinite programming [4]).

C. Yield Optimization Using Device Operating Constraints

Device operating constraints (DOCs) [5], [18] are topology-specific constraints to ensure that devices are operating in the intended region (e.g., transistor must be in saturation), and building block behavior is as expected (e.g., currents in current mirrors must match). References [5] and [6] found that yield using DOCs in optimization is significantly better than yield not using DOCs. References [23] and [24] show that using them within the context of a yield optimizer will improve the optimizer's convergence.

D. Design Centering in Feasibility Region

In this approach, the optimizer aims to find a design point $\boldsymbol{d}$ that shifts the performance contours in the process space S (and therefore the feasible region) to align favorably with the fixed distribution $pdf(\boldsymbol{s})$. Fig. 2 shows the design-centering view of yield optimization.

One variant of design centering models each performance's feasibility δ_i as a linear classifier $\psi_i : \widehat{\delta}_i = \psi_i(\boldsymbol{d}, \boldsymbol{s}, \boldsymbol{\theta})$. Each classifier is built from a sensitivity analysis and SPICE simulations. The linear models are concatenated to form an approximation of the overall feasibility region $\widehat{\boldsymbol{\delta}} = \bigcap_i \{\psi_i(\boldsymbol{d}, \boldsymbol{s}, \boldsymbol{\theta})\}$. By definition, $\widehat{\boldsymbol{\delta}}$ is a convex polytope. Using $\widehat{\boldsymbol{\delta}}$, the algorithm finds a sizing that shifts the polytope approximation to align "favorably" with the fixed $pdf(\boldsymbol{s})$. The algorithm then repeats with further sensitivity analyses. "Favorable" can be 1) maximum worst-case distance from the center of the probability density function (pdf) to the closest feasibility boundary [25] or 2) maximum yield [7], [23], i.e., maximum volume under the pdf that is in the polytope feasible region.

Another variant [8], [26] views $\hat{\boldsymbol{\delta}}$ as an ellipsoid rather than a convex polytope, then aims to maximize the volume of the ellipsoid. The final design is the ellipsoid's center.

A drawback of this approach is that linear models have very poor accuracy in modeling circuits on modern processes, which means that the convex polytope approach will lead to suboptimal designs.

E. Nominal Tradeoffs

This approach does *nominal* multiobjective optimization, followed by local yield optimization from each Pareto-optimal design [9], [10], [12]. Unfortunately, the approach relies upon a tight correlation between nominal and robust designs, which may not be the case in practice as Section II-B discussed.

The next two sections describe two foundational technologies for our SANGRIA solution to address the globally reliable variation-aware sizing of analog circuits.

F. Past Approaches Using MBO

The idea in MBO [22] is to build response surface models on-the-fly during each iteration of optimization, and to optimize on the regression models to propose new designs. In [12], a kriging model taking both design variables and process variables as inputs was used. In [11], a projection-based polynomial was used. One problem with these approaches is blind spots—because of few samples in a design region, the macromodel thinks that the region is poor, whereas, in reality, the region is good. This can cause convergence to a local optimum. The other issue is the specific modeling choices: Kriging models have very poor scaling in the number of input variables ([12] had < 10 design variables), and the projection-based model makes strong assumptions about the nature of the mapping (quadratic).

G. Density Estimation

Some of the approaches above [10], [27] get help from density estimation. Given a small number of SPICE-simulated MC samples at design point $\boldsymbol{d}$, density estimation approximates the pdf across the performances space, $\widehat{pdf}(\boldsymbol{\lambda})$, then estimates yield by

$$\widehat{Y_{\mathrm{DE}}}(\boldsymbol{d}) = E\left\{ \delta(\boldsymbol{d}, \boldsymbol{\lambda}) | \widehat{pdf}(\boldsymbol{\lambda}) \right\} = \int_{\boldsymbol{\lambda} \in \Re^{N_g}} \prod_{i=1}^{N_g} \delta_i\left(\lambda_i(\boldsymbol{d})\right) * \widehat{pdf}(\boldsymbol{\lambda}, \boldsymbol{d})\, d\boldsymbol{\lambda} \quad (11)$$

where $\delta_i(\lambda_i(\boldsymbol{d}))$ is one if λ_i is feasible, and zero otherwise. Yield estimates from $\widehat{pdf}$ can be more accurate than binomial count-based yield estimates [(8)]. Unfortunately, several performance metrics mean that many simulations are needed for an accurate $\widehat{pdf}$. Furthermore, estimates of $pdf(\boldsymbol{\lambda})$ make strong assumptions about the nature of the distribution. For example, the approach [10] finds ten random points "which make the distribution the most Gaussian," or the approach [27] models the frequency distribution as a linear-time-invariant system, which makes the pdf have ringing at sharp drop-offs in density.

IV. Foundations: Homotopy

Homotopy or continuation methods ([14, Sec. 11.3]) are an optimization strategy in which the original optimization problem of solving $f(\boldsymbol{d}) = 0$ is not solved directly. Instead, an easy problem is first set up. This easy problem is gradually transformed to the true problem, and during the transformation, the solution to the problem is continuously tracked. Eventually, the problem has become the true problem, and therefore its solution is the true solution.

We now give an example where the solution to the initial problem can be trivially set as $\boldsymbol{a}$. Define the *homotopy map* $H(\boldsymbol{d}, \eta)$ as

$$H(\boldsymbol{d}, \eta) = \eta * f(\boldsymbol{d}) + (1 - \eta) * (\boldsymbol{d} - \boldsymbol{a}) \qquad (12)$$

where η is a scalar parameter and $\boldsymbol{d}$, $\boldsymbol{a} \in \Re^{N_d}$. When $\eta = 0$, (12) becomes the easy initial problem $H(\boldsymbol{d}, \eta) = \boldsymbol{d} - \boldsymbol{a}$; and therefore when solved to $H(\boldsymbol{d}, \eta) = 0$, $\boldsymbol{d}$ takes the value of the easy initial solution $\boldsymbol{a}$. $H(\boldsymbol{d}, \eta)$ becomes the original problem $f(\boldsymbol{d})$ when $\eta = 1$. The steps in between, i.e., the path in the space of $\boldsymbol{d} \cup \eta$ where $H(\boldsymbol{d}, \eta) = 0$ for various values of η, is called the *zero path*.

There are various strategies for shifting from the easy problem at $\eta = 0$ to the true problem at $\eta = 1$. The most obvious one is to gradually change η from zero to one, and solve at each step along the way. However, this may not always work because the zero path may not always follow monotonically increasing values of η. More successful strategies track the zero path itself, rather than the η value. However, we must note that even the more successful strategies can get stuck at local optima on the path, needing to backtrack to looser objective functions or to different design regions.

We will use the concept of homotopy in SANGRIA—the novel *structural homotopy* approach. Note that in SANGRIA, the easiest problems will not be trivial like the example here.

V. Foundations: MBO

A. MBO Description

Despite limitations of current MBO approaches (Section III-F), MBO is promising, so we describe it further.

To find a design $\boldsymbol{d}$ that maximizes f, MBO works as follows. Its initialization step does space-filling sampling in the design space, e.g., with Latin Hypercube Sampling [28]. In each iteration, the new design(s) are evaluated on f, a model $\hat{f}$ is built, and inner optimization on the model is performed to find a new design x_{guess}. The inner optimization uses an "infill criterion" objective function that combines maximizing $\hat{f}$ and maximizing the model's uncertainty (to identify blind spots).

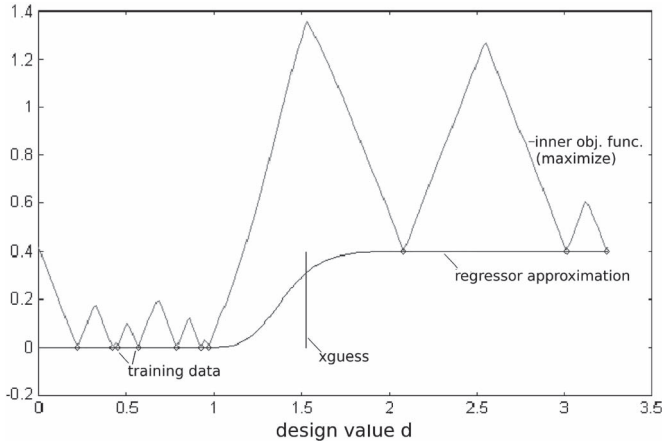

Fig. 3. MBO at the first iteration. There is a single design variable $\boldsymbol{d}$. Each diamond is a training datapoint $\{\boldsymbol{d_j}, f_j\}$. The x_{guess} is chosen by maximizing the infill criterion (mountainlike curve).

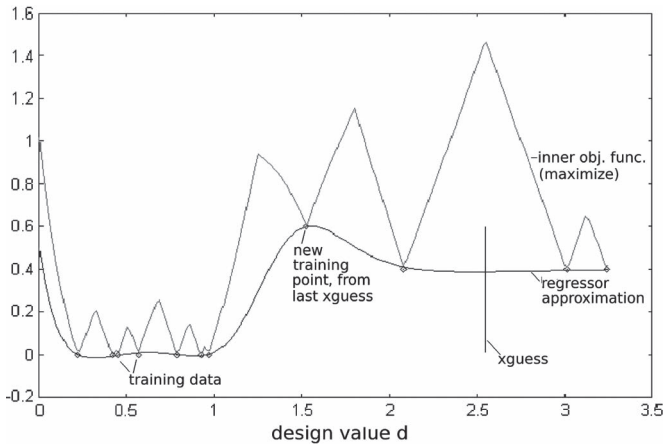

Fig. 4. MBO at the second iteration, after the first iteration's x_{guess} was simulated and the regressor was updated, which uncovered a new optimum. Merely maximizing $\hat{f}(\boldsymbol{d})$ would have missed it.

Few MBO approaches account for model optimality and uncertainty, while modeling the global design space. A notable exception is [22]. It uses a kriging regression model which naturally reports prediction uncertainty. Building on it, [29] tests various infill criteria approaches, and found that the "least-constrained bounds" (LCB) criterion gave the most reliable MBO convergence.

Figs. 3 and 4[2] show two iterations of MBO on a simple 1-D problem. Here, model uncertainty is merely the distance to the closest training point[3]: $u(\boldsymbol{d}) = \min\{abs(\boldsymbol{d} - \boldsymbol{d_1}), abs(\boldsymbol{d} - \boldsymbol{d_2}), \ldots\}$. The LCB infill criterion is $\Lambda(\boldsymbol{d}) = (1 - w_{\text{explore}}) * \psi(\boldsymbol{d}) + w_{\text{explore}} * u(\boldsymbol{d})$, where w_{explore} is the relative weight for exploration compared to exploitation; $w_{\text{explore}} \in [0, 1]$. Since uncertainty is a function of the distance to the closest training point, the LCB curve gets a mountainlike shape on top of the regressor's curve.

[2]The regressor for this illustration is a neural network [30].

[3]Using distance is just *one* way to compute uncertainty; more on this later.

B. MBO Shortcomings

MBO algorithms are promising because they make maximum use of available data. However, the versions in the literature have several issues.

1) Inadequate Regressors: The typical regressor, kriging, scales very poorly with the number of input dimensions and samples. While quadratic-based MBOs like [31] scale better, they only manage to circumvent the nonflexible structure of quadratics by limiting their application to *local* search, whereas we want to do global search for reasons discussed in Section II-B.

2) Issues in Uncertainty: Most regressors do not have a natural way to compute uncertainty. Linear models do, but do not handle nonlinearity. Kriging and density estimation compute uncertainty, but scale poorly. A *regressor-independent* technique is to compute uncertainty as a function of the Euclidian distance from the closest training point(s), as the example in Section V-A described, and with LCB leads to the "mountains." This is fine for a few dimensions, but past 10–15 dimensions, the Euclidean measure is ineffective because all points are very far from each other [32].

3) Sensitivity of Infill Criterion: While LCB is relatively robust compared to expected improvement [22] and other criteria [29], it still shows sensitivity to its w_{explore} setting [29]. We do not want a poor w_{explore} to constrain the ability to perform efficient search and effectively escape local optima.

4) Too Few Samples for High-Dimensional Prediction: Even if we overcome the other issues, if the number of design variables $\geq$ 50 dimensions, and the number of simulations is limited, there is simply too little data to make any meaningful prediction at all. In such cases, MBO will degenerate to random search.

VI. SANGRIA Algorithm

Now that we have described some foundations of SANGRIA—homotopy and MBO—we are prepared to describe SANGRIA itself. We first its high-level structure, then its high-level algorithm, and finally present the details.

A. High-Level Structure

Fig. 5 shows the structure of SANGRIA. Its key elements are structural homotopy and high-dimensional MBO.

1) Structural Homotopy: A set of search layers approximates the exploration-versus-exploitation spectrum; all layers conduct search simultaneously. The lowest layer has the loosest objective function (which happens to be cheaper to evaluate). The intermediate levels refine and further test promising candidates from lower layers, and the top level has the full objective function to thoroughly evaluate the most promising designs. New randomly generated designs are continually fed into the lowest (loosest) layer, which enables SANGRIA to keep trying new design regions and therefore avoid getting stuck in local optima.

2) High-Dimensional MBO: MBO uses training samples from past MBO candidates and from structural homotopy. It uses SGB [15], which handles arbitrary nonlinearities and has excellent predictive ability even in high dimensionality. *Ensembles* of SGB models allow computation of model uncertainty $u(\boldsymbol{d})$ as the standard deviation across the SGBs' outputs. The sensitivity of LCB is resolved by replacing a single-objective optimizer on the infill criterion with a multiobjective optimizer Nondominated Sorting Genetic Algorithm-II [16], which maximizes $f(\boldsymbol{d})$ and $u(\boldsymbol{d})$. If there is still insufficient data for predictive models in MBO, the structural homotopy side of search still drives the search forward, i.e., the SANGRIA algorithm does not *need* MBO, but if MBO suggests useful design candidates, then the search can exploit them.

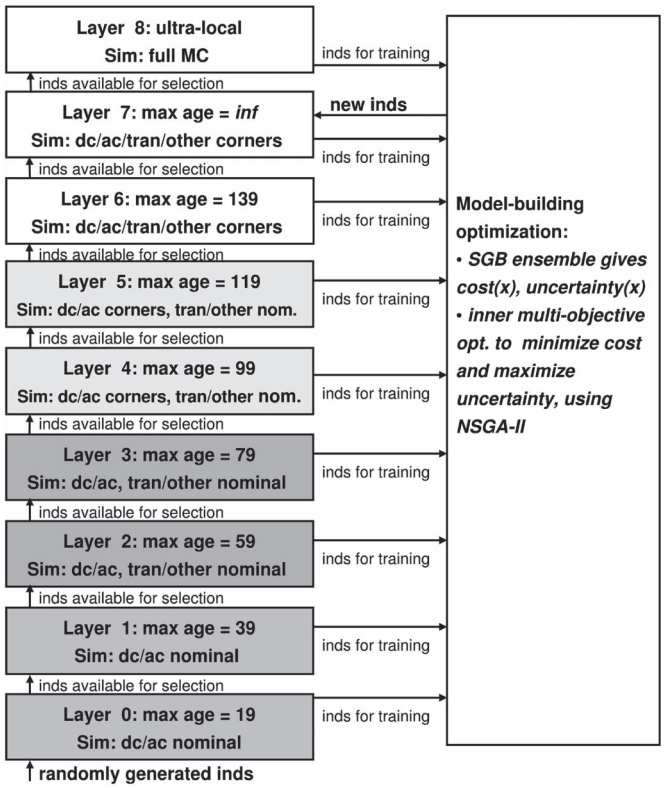

Fig. 5. SANGRIA structure. The left half has structural homotopy, where all layers search simultaneously. Randomly generated designs are continually fed into the (cheap) lowest layer, and higher layers refine the designs. The right half does MBO to efficiently uncover the design-to-objective mapping. Structural homotopy and MBO share search information, each side using the other side's strengths to overcome its own weaknesses.

Each structural homotopy layer is an evolutionary algorithm (EA) to optimize a population of candidate designs ("individuals"), as shown in Fig. 5 (left). The layers are organized according to the degree to which the candidate designs have been optimized ("age"), i.e., an age-layered population structure (ALPS) [33]. Randomly drawn designs enter the lowest layer as zero-age designs, and if they do well, they get promoted to ever-higher layers while being further optimized (and aging $+1$ unit per generation). Each layer has a maximum age: 10 for layer 0, 20 for layer 1, etc. (giving some chance to improve, but not *too* much to stagnate with; similar to [33]). If a design gets too old for a given layer, then it is ejected from that layer, thereby preventing wasted search effort on a stagnated design.

Other homotopy algorithms work by starting with an easier-to-solve loosened version of the problem, then tightening the problem *dynamically*. In contrast, *structural* homotopy embeds the loosening into the algorithm's data structure (state). The

solution to each layer can be regarded as a point on the homotopy "zero path." Therefore, we can view structural homotopy as a new approach to traverse the zero path: Learn it coarsely to begin with, and refine it over time. This gives structural homotopy a useful advantage over traditional homotopy. Traditionally, if homotopy converges locally while on the zero path, backtracking must be employed and a new branch of the zero path explored. In contrast, structural homotopy sees several regions at once, only refining the promising regions.

Specifically, as shown in Fig. 5, layer 0 is just simulated at a single process/environmental corner of {dc/ac analyses, nominal process point $\boldsymbol{s}$, typical environmental point $\boldsymbol{\theta}$}. Layer 1 is like layer 0. Then, layer 2 adds transient/other analyses on the single $\{\boldsymbol{s}, \boldsymbol{\theta}\}$ corner. Layer 4 adds nonnominal corners for dc/ac, and layer 6 adds nonnominal corners for transient/other. The choice of corners is elaborated in Section VI-G. Finally, layer 8 does a full MC simulation (with blocking) on each candidate. This split of simulations was chosen based on choosing analyses which give many key measures for less simulation cost (ac, dc), and deferring the more expensive analyses which only give incremental measures of quality (transient and corners). The core idea of structural homotopy does not depend on the exact choice just presented, however; an alternative would be to always simulate all testbenches but have more MC samples at higher levels. The number of layers is flexible as well—the core aim is to approximate continual tightening of the problem, and discretization to nine layers of tightening is reasonable; a few more or a few less is fine too.

SANGRIA's lower layers have larger populations which shrink going upwards. This balances out the simulation cost per age layer, encourages more exploration at the lower levels, and reduces expensive top-layer simulation costs. The top layer has a tiny population, hence the label "ultralocal."

Each layer follows an EA framework for updating the population with selection operators and search operators. Selection for layer i is typical EA selection, except that individuals from layer $i-1$ are also considered.

SANGRIA's search effectiveness is due to structural homotopy, MBO, and their combination. Structural homotopy allows continual coverage of the whole exploration-versus-exploitation, cheap exploration, and a natural avoidance of local optima. MBO improves efficiency because new candidate designs can be selected more intelligently. The combination means that the advantages of both structural homotopy + MBO can be exploited (exploration + efficiency), while reducing their disadvantages if each were standalone (lower efficiency + poor prediction if few samples).

B. SANGRIA High-Level Algorithm

SANGRIA's high-level algorithm, *SangriaOptimization()*, is described in Table II. The algorithm's inputs are the search space bounds D, age gap N_a (described later), maximum number of layers K, and number of individuals $N_L(k)$ for each layer k, and it outputs the optimal design $\boldsymbol{d}^*$.

Line 1 initializes: the generation count, N_{gen}; the data structure P which will hold a population at each age layer P_k; and all individuals encountered so far in the search, P_{all}. Lines 2–13 are the generational loop, which repeats until stopping conditions are met.

TABLE II
PROCEDURE SANGRIAOPTIMIZATION()

Inputs: D, N_a, K, $N_L(k)$
Outputs: $\boldsymbol{d}^*$
1. $N_{gen} = 0$; $P = \emptyset, P_{all} = \emptyset$
2. while stop() $\neq True$:
3. if $(N_{gen} \% N_a) = 0$:
4. if $|P| < K$:
5. $P_{|P|+1} = \emptyset$
6. P_0 = SpaceFillIndividuals($N_L(k), N_D, D$)
7. for k = 1 to $|P|$:
8. P_k = SelectParents($P_k, P_{k-1}, N_L(k)$)
9. $P_{k,j}$ = UpdateLocalOptState($P_{k,j}, k$), j = 1 to $|P_k|$
10. P_{all} = unique($P_{all} \cup P$)
11. $P_{|P|} = P_{|P|} \cup$ InnerOptimize(P_{all}, D, k)
12. $\boldsymbol{d}^*$ = $\boldsymbol{d_i}$ in P_{all} with highest Y or Cpk
13. $N_{gen} = N_{gen} + 1$
14. return $\boldsymbol{d}^*$

Lines 3–6 handle the case of an "age-gap" generation which happens every N_a generations. In an age-gap generation, the zeroth layer gets $N_L(0)$ new space-filling individuals in the N_D-dimensional space D, including a "loose" layer-0 evaluation. Space-filling sampling uses Latin Hypercube Sampling [28] with uniform distribution across the whole design space D. P starts out with just one layer. At the first "age gap" generation, it grows a new layer. At each subsequent "age gap" generation, it adds a new layer, until steady state with K layers as Fig. 5 shows ($|P| = K$). MBO always feeds to the current top (nonultralocal) layer $P_{|P|}$.

In lines 7–9, each age layer P_i is updated. First, acceptably young parents are selected from the current or next lower layer. Then, each individual's local state χ is updated, including evaluations appropriate to the age layer k (in line with structural homotopy). Line 10 updates all the individuals encountered so far, P_{all}, just in time for the MBO inner optimization (line 11). For efficiency, if a layer has solved all its constraints, it skips the call to *UpdateLocalOptState()* for that layer.

Lines 12 and 13 update the best design so far $\boldsymbol{d}^*$ and the generation count N_{gen}, respectively. When the search terminates, $\boldsymbol{d}^*$ is returned; and of course, during search, intermediate $\boldsymbol{d}^*$s can be returned.

The following sections elaborate on SANGRIA details.

C. SANGRIA Individuals

The atomic unit that SANGRIA processes is an "individual." In most EAs, an individual is a single design candidate, and new designs are generated through mutation or crossover operators. Unfortunately, those operators are slow because they do not exploit the past information about search. Memetic EAs run a local optimization as part of each individual's evaluation, but it is unclear how much optimization effort should be given to each individual.

For efficient EA search, each SANGRIA individual is a local optimization search *state* which takes exactly one step per generation. Therefore, it exploits past information about search, without the difficulties of memetic EAs. The search state χ holds: 1) one or more design points; 2) associated circuit

TABLE III
PROCEDURE SELECTPARENTS()

Inputs: $P_k, P_{k-1}, N_L(k)$
Outputs: P'_k
1. P_{cand} = ageOk($P_k \cup P_{k-1}$)
2. for i = 1..$N_L(k)$:
3. $\quad par1 \sim unif(P_{cand})$
4. $\quad par2 \sim unif(P_{cand})$
5. $\quad P'_{k,i}$ = best($\{par1, par2\}$)
6. return P'_k

evaluations; and 3) local optimizer-specific state information such that each individual's local optimization can be paused and restarted each generation.

D. Local Optimization Search Operator

The local optimizer is an efficient derivative-free algorithm called dynamic hill climbing (DHC) [34]. DHC is a hillclimber; when it finds improvements, it capitalizes on the direction of improvement with acceleration and ridge walking.

DHC was chosen for a few reasons. First, derivatives are costly to compute, which rules out classical nonlinear programming algorithms such as quasi-Newton with Broyden–Fletcher–Goldfarb–Shanno update [14]. Second, the search space has discrete variables, ruling out many modern derivative-free algorithms such as NEWUOA [31]. Nature-inspired algorithms such as EAs, simulated annealing, and particle swarm optimization are derivative free and can handle continuous or discrete spaces, but have a global (not local) focus. Pattern search algorithms [35], [36] are derivative free, can handle mixed spaces, and have a local search focus. These are reasonable choices, and have been used in other sizers [6]. DHC can be viewed as a loosened version of pattern search—loosened because it allows for step-size growth in order to improve convergence rate, at the expense of losing some theoretical properties of pattern search convergence. Since we have many local optimizers in parallel, we are less concerned about provable convergence per local optimizer, and more concerned with convergence *rate*; hence, we chose DHC in SANGRIA.

SANGRIA only sees that the (DHC) individual offers a design point ($\boldsymbol{x}$), an associated cost for that point, and a routine to update the individual's local optimization state *updateLocalOptState()*, which alter χ_{DHC} according to [34].

E. ALPS Selection

Table III describes tournament selection of parents in SANGRIA. Line 1 determines the candidate parents P_{cand} by merging layer k and layer $k-1$, and only keeping the individuals with age $\leq$ maximum age at layer k. Lines 2–5 fill the selected population: Lines 3 and 4 randomly draw parents 1 and 2 with uniform bias from P_{cand}, and line 5 selects the parent with the lowest cost. Line 6 returns the updated population P'_k.

F. SANGRIA MBO

This section describes how MBO is deployed within SANGRIA. Table IV describes the high-level MBO algorithm *InnerOptimize()*. Lines 1 and 2 build the training input and output data, respectively, using the information of all the individuals so far, P_{all}. $P_{\mathrm{all},1}$ is the first individual in this list of all individuals, $P_{\mathrm{all},2}$ is the second, and so on. $P_{\mathrm{all},1}.\boldsymbol{d}$ is the design point of the first individual, and so on.

TABLE IV
PROCEDURE INNEROPTIMIZE()

Inputs: P_{all}, D, k
Outputs: P_{inner}
1. $\boldsymbol{X} = \{P_{all,1}.\boldsymbol{d}, P_{all,2}.\boldsymbol{d}, \dots\}$
2. $\boldsymbol{y} = \{cost(P_{all,1}, k), cost(P_{all,2}, k), \dots\}$
3. ψ = BuildSgbEnsemble($\boldsymbol{X}$, $\boldsymbol{y}$, N_{ens})
4. $P_{inner} = \left\{ \begin{matrix} minimize\{cost(\psi, \boldsymbol{d})\} \\ maximize\{u(\psi, \boldsymbol{d})\} \end{matrix} \right\}$ s.t. $\boldsymbol{d} \in D$
5. P_{inner} = cluster(P_{inner}, N_{inner})
6. return P_{inner})

TABLE V
PROCEDURE EVALUATE()

Inputs: P_k, k, K
Outputs: P'_k
1. for i = 1..$|P_k|$:
2. $\quad$ simulate $P_{k,i}$ for layer $min(k+1, K)$ specifications
3. $P'_k = P_k$; return P'_k

Line 3 constructs the regressor ψ, an SGB ensemble, from the training data $\{\boldsymbol{X}, \boldsymbol{y}\}$. In line 4, an inner optimization is run according to the problem formulation. Since there are two objectives (rather than a single sensitive infill criterion), a Pareto-optimal set of designs is returned to collectively approximate ψ's exploration–exploitation tradeoff. The multiobjective optimization is performed using NSGA-II [16].

Multiobjective optimization could return a large number of Pareto-optimal individuals. We do not want to evaluate all of these because it could become computationally expensive. Therefore, line 5 reduces the number of individuals from $|P_{\mathrm{inner}}|$ to N_{inner}, using clustering. SANGRIA employs bottom-up clustering (hierarchical agglomerative clustering) [37].

G. Setting Corners

SANGRIA's objectives are computed by measuring performance on a set of corners which are set at the beginning of the run. Recall that the core idea of corners-based approaches is as follows: If corners are "representative" of process and environmental variations, and all corners can be "solved," then the final design's yield will be near 100% [(10)].

The challenge is to choose corners that are representative of the performance bounds, with a minimum count, and without any assumptions on the mapping from process variables to performance. SANGRIA's approach is to: 1) take $N_{\mathrm{MC,cand}}$ (e.g., 100) samples of process points, simulate them all at a typical environmental point, then 2) choose $N_{\mathrm{MC,chosen}}$ (e.g., 25) representative points (corners). Representative corners are chosen in two steps: 1) Do nondominated filtering toward worst performance values, i.e., nondominated filtering in the opposite directions of optimal, and 2) if needed, further reduce the points by bottom-up clustering [37].

H. Evaluation and Cost Calculation

Table V describes the evaluation of a population at age layer k, P_k. Each design candidate $\boldsymbol{d}$ at layer k must be

evaluated sufficiently for use in selection at layer k and at layer $k+1$ (line 2). The $\min()$ accounts for the top (Kth) layer. SANGRIA's per-layer simulation specifications are shown in Fig. 5 (left). For example, layer 2's specification is {dc/ac nominal, transient/other nominal}. Therefore, layer-1 individuals must also be simulated at those specifications, as its individuals are available for selection in layer 2.

When an individual is evaluated "on nominal," each of its DHC state's $\boldsymbol{d}$s are simulated at {nominal process point $\boldsymbol{s_{nom}}$, typical environmental point $\boldsymbol{e_{typ}}$}. When evaluated "on corners," it means that the evaluated is simulated at: 1) all representative $\boldsymbol{s}$s with $\boldsymbol{e_{typ}}$; and 2) all $\boldsymbol{e}$s with $\boldsymbol{s_{nom}}$. This avoids simulating *all* combinations of environmental and process points. Then, the performance λ at a given $\{\boldsymbol{d}, \boldsymbol{s}, \boldsymbol{e}\}$ is estimated as the performance at $\{\boldsymbol{s_{nom}}, \boldsymbol{e_{typ}}\}$, summed with deltas in performance due to $\boldsymbol{s}$ and $\boldsymbol{e}$

$$\begin{aligned}\hat{\lambda}(\boldsymbol{d}, \boldsymbol{s}, \boldsymbol{e}) = &\lambda(\boldsymbol{d}, \boldsymbol{s_{nom}}, \boldsymbol{e_{typ}}) + (\lambda(\boldsymbol{d}, \boldsymbol{s}, \boldsymbol{e_{typ}}) \\ &- \lambda(\boldsymbol{d}, \boldsymbol{s_{nom}}, \boldsymbol{e_{typ}})) + (\lambda(\boldsymbol{d}, \boldsymbol{s_{nom}}, \boldsymbol{e}) \\ &- \lambda(\boldsymbol{d}, \boldsymbol{s_{nom}}, \boldsymbol{e_{typ}}))\,. \end{aligned} \tag{13}$$

When the algorithm estimates the cost of an individual, the layer k is important. For example, an individual may have enough simulations for layer 2, but is participating in a layer-1 selection tournament; then, its cost calculations only need to use the simulations that layer 1 specifies. The cost is computed as follows:

$$cost(\boldsymbol{d}) = cost_g(\boldsymbol{d}) + cost_{\text{cpk}}(\boldsymbol{d}) \tag{14}$$

where $cost_g$ measures the total cost of violating constraints and $cost_{\text{cpk}}$ is a contribution from measuring Cpk

$$cost_g(\boldsymbol{d}) = \sum_{i}^{N_g} violation\left(\widehat{g_{\text{wc},i}}(\boldsymbol{d}, \lambda_i)\right) \tag{15}$$

$$violation(g_i) = \begin{cases} 0, & g_i \le 0 \\ \frac{g_i - g_{i,\min}}{g_{i,\max} - g_{i,\min}}, & \text{otherwise} \end{cases} \tag{16}$$

where $\widehat{g_{\text{wc},i}}$ is the estimated worst-case value of performance i across all $\{\boldsymbol{s}, \boldsymbol{d}\}$ combinations. Performance is estimated at each $\{\boldsymbol{s}, \boldsymbol{d}\}$ combination with (13). $g_{i,\max}$ and $g_{i,\min}$ are the minimum and maximum values of performance g_i seen so far in the optimization run.

The additional $cost_{\text{cpk}}$ is activated when all constraints are solved, and pulls cost < 0 depending on how high the Cpk is. It enables the optimizer to increase the margin further, once the estimated yield hits 100%

$$cost_{\text{cpk}}(\boldsymbol{d}) = \begin{cases} 0, & cost_g(\boldsymbol{d}) = 0 \\ -(Cpk(\boldsymbol{d}) + cpk_{\text{off}})\,, & \text{otherwise} \end{cases} \tag{17}$$

where cpk_{off} is a value sufficiently large to ensure that negative values of Cpk do not make the overall value of cost be > 0. Cpk is calculated with (5).

TABLE VI
TEST CIRCUIT SIZES

Circuit	Num. Devices	Num. Design Vars.	Num. Process Vars.	Num. Env. Vars.	Num. Env. Points	Test-benches
10T opamp	10	21	91	5	3	ac, tran, THD
30T opamp	30	56	216	5	3	ac, tran, THD
50T opamp	50	97	342	5	3	ac, tran, THD
vref	12	28	106	3	3	ac, ac

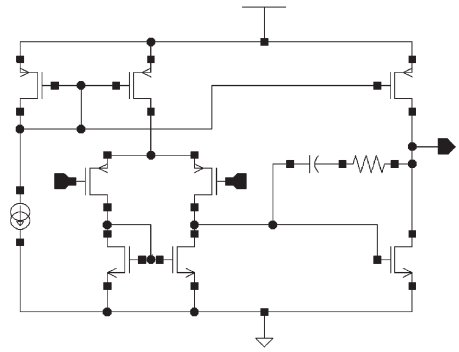

Fig. 6. Schematic of ten-device operational amplifier.

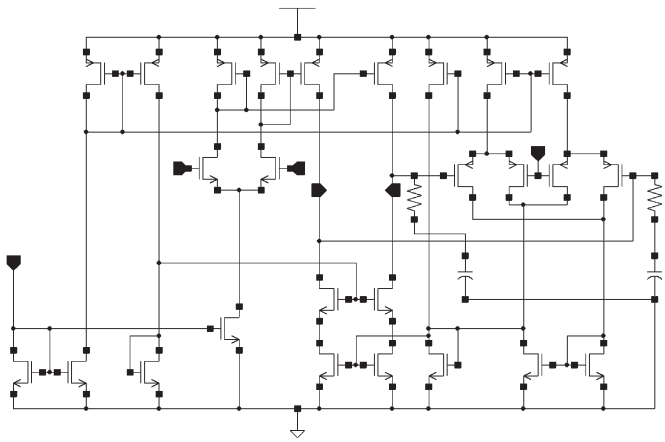

Fig. 7. Schematic of thirty-device operational amplifier.

VII. SANGRIA EXPERIMENTAL RESULTS

A. Circuit Problems

We used the test circuits outlined in Table VI and shown in Figs. 6–9, which includes three opamps of increasing size (from 10 to 50 devices), and a voltage reference ("vref"). For each circuit, we performed several runs with different seeds to the random number generator. We will analyze the results of all runs.

Fig. 6 shows the 10T opamp. Specifications were: gain $A_V >$ 65 dB, bandwidth $BW > 1$ MHz, gain bandwidth $GBW >$ 300 MHz, phase margin $PM > 56°$, gain margin $GM <$ -10 dB, settling time $ST < 12$ ns, slew rate $SR > 3e8$ V/s, overshoot $OS < 12\%$, and total harmonic distortion $THD <$ -45 dB.

Fig. 7 shows the 30T opamp. Specifications were: $A_V >$ 37.5 dB, $BW > 13.5$ MHz, $GBW > 300$ MHz, $PM > 59°$,

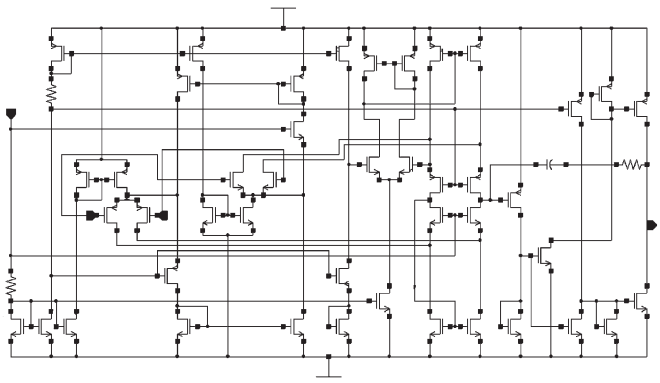

Fig. 8. Schematic of fifty-device operational amplifier.

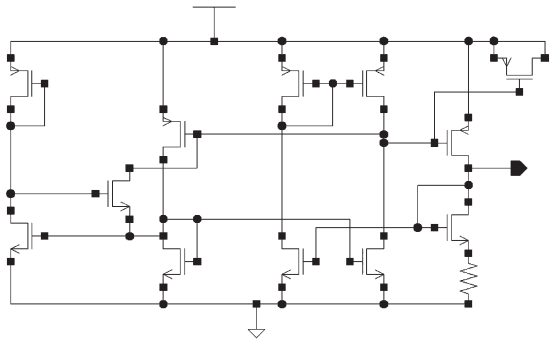

Fig. 9. vref schematic.

$GM < -10$ dB, unity gain frequency $FU > 265$ MHz, $ST < 5$ ns, $SR > 1.85e8$ V/s, $OS < 6\%$, and $THD < -40$ dB.

Fig. 8 shows the 50T opamp. Specifications were: $A_V > 30$ dB, $BW > 2.3$ MHz, $GBW > 50$ MHz, $PM > 65°$, $GM < -5$ dB, $FU > 50$ MHz, $ST < 15$ ns, $SR > 1.5e8$ V/s, $OS < 5\%$, and $THD < -40$ dB.

Fig. 9 shows the vref. Specifications were: power $PWR < 0.111$ mW, temperature coefficient $TC < -20$ °C, minimum temperature $TMIN < -20$ °C, maximum temperature $TMAX > 85$ °C, voltage-change reference $DVREF < 600$, minimum voltage $VMIN < 0.78$ V, maximum voltage $VMAX > 2.8$ V.

B. Technology and Variation Model

The technology was Taiwan Semiconductor Manufacturing Corporation 0.18-μm CMOS. The simulator was a proprietary SPICE-like simulator of a leading analog semiconductor company, with accuracy and runtime comparable to HSPICE. We used the process variation model of [13] because of its excellent accuracy, and to illustrate the ability of SANGRIA to handle a large number of process variables. Accordingly, the local variation parameters for each transistor are the following: NSUB (substrate doping concentration), VFB (flatband voltage), WINT (width variation), LINT (length variation), U0 (permittivity), RSH (sheet resistance), and TOX (gate oxide thickness). The per-resistor variation parameters are the following: DRSH (sheet resistance), DXW (width variation), and DXL (length variation); and the per-capacitor variation parameters are the following: DXW (width variation), DXL (length variation), and DTOX (oxide thickness). There is a single global-variation parameter for each of NSUB, VFB, etc., as well. The variables $\boldsymbol{s}$ in the process variations' $pdf(\boldsymbol{s})$ are normal, independent, and identically distributed.

C. Algorithm and System Settings

Each run of each circuit problem had identical algorithm parameters. The parameters had little tuning, instead being set based on reasoning, choosing to err on the side of reliability. The maximum number of circuit simulations was $N_{sim,\max} = 100\,000$, which is easy to run overnight with a modestly sized computer cluster. (Therefore, all the runtimes for each forthcoming SANGRIA run are overnight or less.)

Similar to the parameters of ALPS [33], there were $K = 9$ age layers (in line with Fig. 5) with age gap $N_a = 10$. The lowest age layer's population size $N_L(0)$ was 200 individuals. Going to higher layers, the population size decreased linearly from $N_L(0) = 200$ to $N_L(7) = 8$. The ultralocal layer had $N_L(8) = 3$ individuals, which allowed some exploration without being overly computationally expensive. Population sizes of 1–200 are common in EAs. $cpk_{\text{off}} = 10.0$.

In all cases, an initial "rough cut" design is supplied, which took about 10–30 min for an expert designer to do. We do this only so that we can have a baseline to compare the yield and performance spread of initial versus resulting designs. SANGRIA can leverage this, but does not rely on it, because in every $N_a = 10$ generations, it will inject randomly generated designs into age layer 0. $N_{\text{MC,chosen}} = 25$ representative process points were chosen from $N_{\text{MC,cand}} = 100$ candidate points using the approach of Section VI-G.

MBO settings were as follows. SGB parameters were: learning rate $\alpha = 0.10$, minimum tree depth $= 2$, maximum tree depth $= 7$, and target training error $= 5\%$. There were five SGBs in an SGB ensemble. See [15] for details about SGB parameters. NSGA-II parameters were: $N_{\text{pop}} = 25$, $N_{\text{gen,max}} = 50$, with a crossover probability of 0.2. The number of individuals returned from an inner optimization N_{inner} was set to five, which is large enough to get a good spread of the exploration-versus-exploitation tradeoff without becoming too expensive.

Final-result designs (from the optimizer's perspective) had $N_{\text{MC}} = 30$ process points. The lower bound for 100% yield on 30 MC samples is 88.6%, with 95% confidence using Wilson's confidence interval for a binomial proportion [38]. For a more accurate yield estimate, we also report final designs' yield with 2000 MC samples. This also underscores our motivation to make Cpk the objective function rather than yield: Even with 30 MC samples, Cpk means we can meaningfully improve a design when 30/30 MC samples are feasible by increasing the margin and reducing the spread of performances.

D. Experiments on the 10T Opamp Circuit: Run 1 Results

Fig. 10 shows the yield versus generation, and Cpk versus generation for the first run. Each square in the plot is the result of a full MC simulation of the current most promising SANGRIA design across $N_{\text{MC}} = 30$ process points. We see on the far left of the plot that the initial design's yield is 26.7%, and that the next MC sampling happens at generation 60, giving an improved yield of 56.7%. The best yield keeps improving

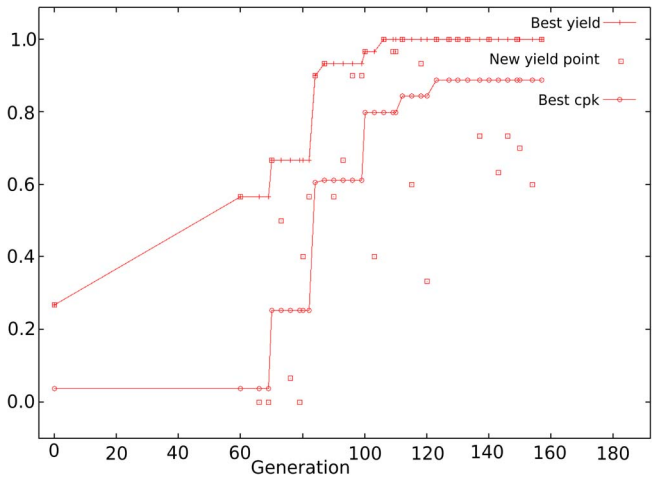

Fig. 10. Best yield versus generation, and best Cpk versus generation, for SANGRIA run 1 on the 10T opamp.

TABLE VII
BEST 10T OPAMP DESIGNS FROM FOUR SANGRIA RUNS

Label	**Yield** (lower, upper)	**Area** (m^2)
Initial design	26.7%	11.60e-10
Run 1 Best	95.75% (94.77, 96.54)	8.88e-10 (-23.4%)
Run 2 Best	99.55% (99.14, 99.76)	10.25e-10 (-11.6%)
Run 3 Best	99.30% (98.83, 99.58)	10.32e-10 (-11.0%)
Run 4 Best	99.95% (99.72, 99.99)	12.04e-10 (+3.80%)

with passing generations, until hitting the maximum of approximately 100% yield (30/30 MC samples) at generation 106.

Since the yield is not precisely estimated, SANGRIA continues to optimize the design using Cpk as the objective, which will implicitly pull up yield as it increases the margin and decreases the spread of performance metrics. Fig. 10 also shows the best Cpk versus generation, denoted by the curve with the ○s. We see that Cpk increases steadily prior to the approximate 100% yield design at generation 106, but it improves further *after* achieving approximate 100% yield. The run stopped when the 100 000 simulation budget was hit. The design with highest Cpk was found in generation 123. (Accurate estimates for all final yield numbers are presented in Table VII).

Note the squares below the curve of yield versus generation. These are MC-sampled results where the candidate design did not do as well as the best so far. It happens when the best design so far on the "ultralocal" layer has already been simulated, so a different design is tried, either from the ultralocal layer or a lower layer.

We can gain insight about SANGRIA's globally reliable characteristic in practice by examining the figures of cost versus generation for each age layer, such as Fig. 11. At generation 0, only the zeroth age layer exists, so only its curve is plotted at first. It was able to immediately meet all the layer-0 constraints (ac/dc nominal), for a cost of 0. At generation 10 (the next age-gap generation), layer 1 is added, and it can fully solve the design as well because it has the same goals as layer 0. At generation 20, layer 2 is added, and despite having more goals (tran/other nominal), it was able to solve them, so its cost stays at zero. At generation 30, the population formerly at layer 2 gets pushed into layer 3. The new individuals going into layer 2 do *not* immediately solve all the goals at generation 30, so their best cost is > 0. In the plot, these are the ○s at a cost value of $\approx$48 for generations 30–33. However, those ○s go back to cost $= 0$ at generation 34 as the new individuals at layer 2 improved the designs.

At generation 40, layer 4 is added and is immediately solved by the individuals coming from layer 3. At generation 50, layer 5 is added, and is solved immediately too. Throughout the whole run, layers 4 and 5 have zero cost. Since the only difference between them and layer 4 is adding corners on the ac testbench, it implies that once a design can solve for the process and environmental variations on ac performances, it can solve for the nominal dc/tran/THD performances. It does not imply that solving on nominal always means solving on corners, however! In fact, we confirm this when layer 6 is added at generation 60, where cost goes > 0.

Layer 8 further meets cost $= 0$ at generation 84. Since it is already considering all the testbenches and process/environmental variations, it starts to aim for cost values < 0. It steadily reduces the cost from generation 84 onwards (the stars curve).

E. Experiments on the 10T Opamp: Results for Runs 2, 3, and 4

We did a second run of SANGRIA on the 10T opamp problem. The run's convergence curves are shown in Fig. 12. It achieved an approximate yield of 100% (30/30 MC samples) at about generation 100. Run 2 illustrates a case of SANGRIA escaping from a local yield/Cpk optimum. We see that the top age layer does not get cost < 0 until generation 110 [Fig. 12 (bottom)]. There was an aborted attempt at generation 70, where the second-highest layer got cost zero, but that design did not translate to the top age layer with low cost. This illustrates that taking steps from the initial design, no matter how promising, might lead to a local optimum. Therefore, there must be an opportunity to try alternative regions. This reconfirms the need to have *globally* reliable statistical optimization.

We did two more subsequent runs of SANGRIA on the 10T opamp problem; each run hit approximate 100% yield (30 MC samples) at about generation 100. The convergence curves had similar profiles to runs 1 and 2. Table VII shows, for each run, the area and yield (on 2000 MC samples). A yield of 99.55% can be achieved while reducing the area by 11.6%. If one is willing to compromise yield to 95.75%, a 23.4% reduction in area is possible. To get the higher yield of 99.95%, area needs to be increased by 3.8%.

F. Experiments on the 30T Opamp Circuit

We performed four independent SANGRIA optimization runs on the 30T opamp. All four runs hit estimated 100% yield on 30 MC samples, and $> 99.0\%$ yield on 2000 MC samples as Table VIII shows. In each run, once 30/30 MC samples were feasible, the run kept improving Cpk significantly beyond.

Each convergence curve shows the signature behavior seen on the 10T problem. The convergence of 30T's run 3 (Fig. 13) is particularly interesting, because it only got good results very late in the run. The lower age layers repeatedly try different regions, until good results are achieved. This reconfirms the value of SANGRIA's age-layered approach to achieving global reliability.

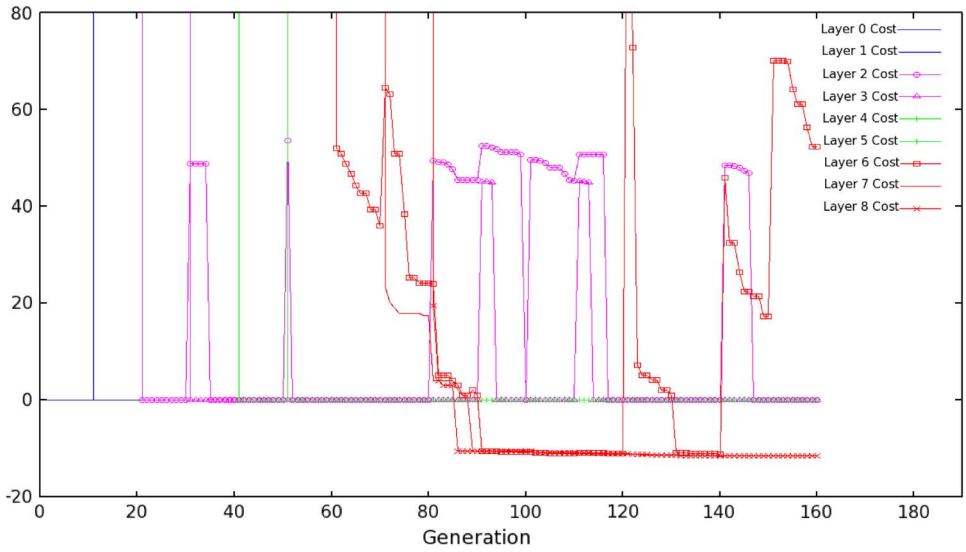

Fig. 11. Best cost versus generation, for each age layer, on SANGRIA run 1 of the 10T opamp.

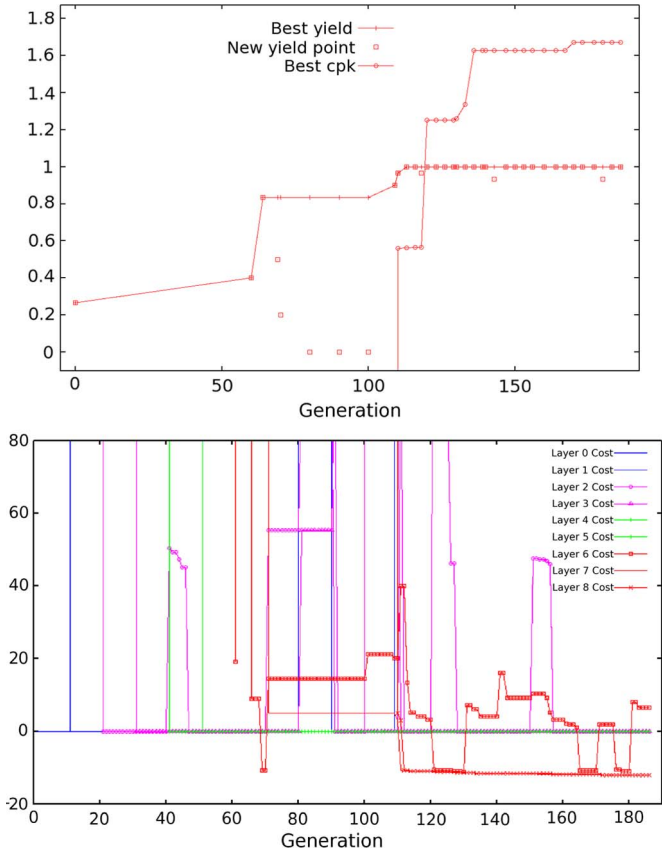

Fig. 12. Convergence curves for SANGRIA run 2 on the 10T opamp. (Top) Best yield/Cpk versus generation. (Bottom) Best cost versus generation.

G. Experiments on the 50T Opamp Circuit

Recall that the 50T opamp has 97 design variables (Ws, Ls, etc.) and 342 process variables. Therefore, these experiments demonstrate the ability of SANGRIA to scale to a very large number of design variables and an even larger number of process variables. The first, third, and fourth runs hit estimated yield of 100% (on 30 MC samples) in under 100 000 simulations, and the second run got close. In the cost-per-layer curves of the second run [Fig. 14 (top)], we see that exploration continues throughout the run. Therefore, just like the user would likely do, we allowed the search to continue farther until it hit the target yield, which it got after 73 further generations (generation 254). This is global reliability: The user does not need to worry about whether the algorithm is stuck at a local optimum. Accurate yield numbers are in Table VIII.

TABLE VIII
SUMMARY OF SANGRIA RESULTS. EACH RUN TOOK < OVERNIGHT

Problem:Run	**# Vars.**	**Init. Yield**	**Final Yield** (lower, upper)
10T amp : 1	117	26.7%	95.75% (94.77, 96.54)
10T amp : 2	117	26.7%	99.55% (99.14, 99.76)
10T amp : 3	117	26.7%	99.30% (98.83, 99.58)
10T amp : 4	117	26.7%	99.95% (99.72, 99.99)
30T amp : 1	277	20.0%	100.00% (99.81, 100.00)
30T amp : 2	277	20.0%	100.00% (99.91, 100.00)
30T amp : 3	277	20.0%	99.15% (98.64, 99.47)
30T amp : 4	277	20.0%	99.20% (98.70, 99.51)
50T amp : 1	444	23.3%	98.40% (97.75, 98.86)
50T amp : 2	444	23.3%	98.90% (98.04, 99.38)
50T amp : 3	444	23.3%	99.10% (98.58, 99.43)
50T amp : 4	444	23.3%	99.00% (98.46, 99.35)
vref : 1	137	16.7%	99.65% (99.30, 99.83)
vref : 2	137	16.7%	99.20% (98.70, 99.51)
vref : 3	137	16.7%	98.85% (98.28, 99.23)
vref : 4	137	16.7%	99.50% (99.05, 99.73)

H. Experiments on the vref Circuit

We performed four independent SANGRIA runs on the vref circuit. All four runs hit estimated yield of 100% on 30 MC samples, and $> 99.0\%$ on 2000 MC samples as Table VIII shows. Once again, each of the per-layer cost convergence curves shows the signature behavior that we examined in detail on the 10T problem.

I. Summary of Results

Table VIII summarizes the yield improvements made by each of the 16 SANGRIA runs across the four different circuit test problems. Final yield is estimated from 2000 MC samples. The (upper, lower) values are 95% binomial confidence intervals.

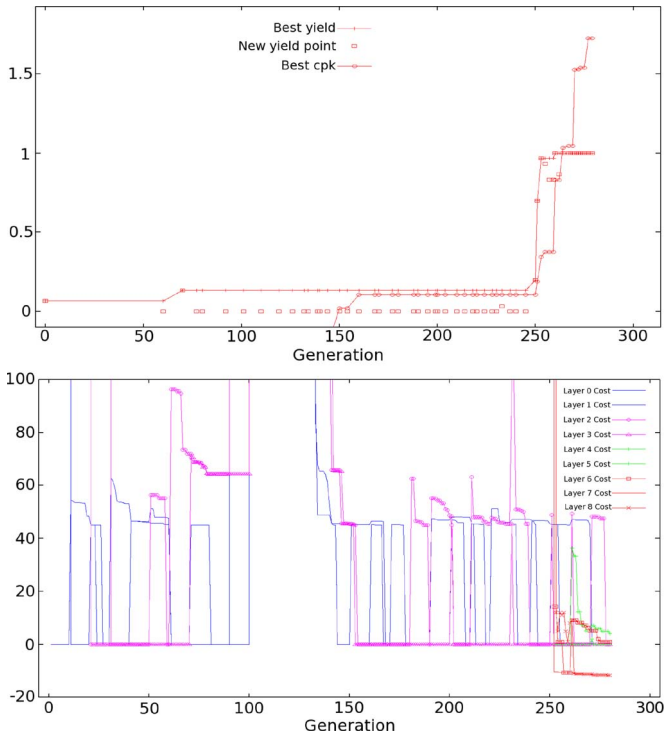

Fig. 13. Convergence curves for SANGRIA run 3 on the 30T opamp. (Top) Best yield/Cpk versus generation. (Bottom) Best cost versus generation.

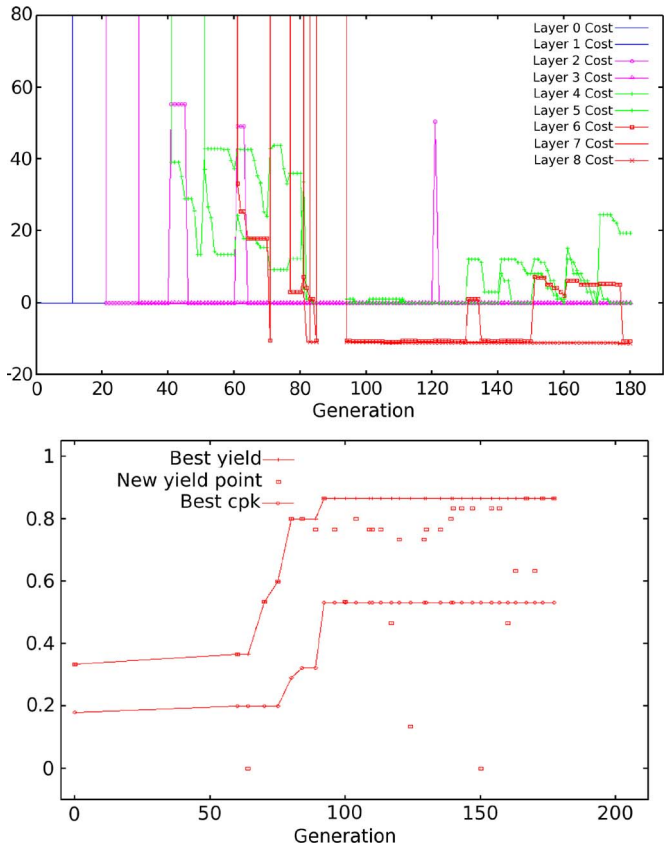

Fig. 14. Convergence curves for SANGRIA run 2 on the 50T opamp. (Top) Best yield/Cpk versus generation. (Bottom) Best cost versus generation.

TABLE IX
SANGRIA SEARCH EFFORT WITH INCREASING PROBLEM SIZE

Problem	# Vars.	# Design Vars.	# Generations For Approx. 100% Yield (Per Run)	Avg. # Gens.
10T amp	117	21	106, 113, 110, 69	99.5
vref	137	28	70, 86, 110, 145	102.75
30T amp	277	56	80, 60, 260, 136	126.5
50T amp	444	97	112, 254, 184, 130	170.0

Table IX shows the effect of the problem size (number of variables) on the overall search effort (number of generations to hit a design with 100% yield on 30 MC samples). Going from the 10T to the 30T problem (2.5× more variables), the search effort only increases by 1.3× on average. Going from the 10T to the 50T problem (4× more variables), search effort only increases by 1.7× on average.

VIII. CONCLUSION

This paper has thoroughly specified the analog circuit variation-aware sizing problem, then reviewed the existing approaches. No approach had the combination of: 1) an accurate variation model; 2) ability to escape local yield/Cpk optima; 3) handling nonconvex/discontinuous mappings; and 4) good scaling with more design and process variables. Then, this paper has presented SANGRIA, which possesses characteristics 1)–4). SANGRIA's key elements are structural homotopy and improved MBO including scalable SGB regression models.

We have tested SANGRIA on four different circuit problems ranging from 10 to 50 devices with a highly accurate process variation model, having up to 444 variables, and several runs per circuit. In all 16 runs, SANGRIA was able to attain near 100% yield and improve the margin within an industrially feasible number of simulations and runtime, despite the high parameter count and the evidence of multimodality.

REFERENCES

[1] International Technology Roadmap for Semiconductors, last accessed Apr. 2008. [Online]. Available: http://public.itrs.net
[2] J. Rattner, "EDA for digital, programmable, multi-radios," in *Proc. Des. Autom. Conf.*, Anaheim, CA, Jun. 10, 2008.
[3] G. G. E. Gielen, W. Dehaene, P. Christie, D. Draxelmayr, E. Janssens, K. Maex, and T. Vucurevich, "Analog and digital circuit design in 65 nm CMOS: End of the road?" in *Proc. Des. Autom. Test Eur. Conf.*, 2005, pp. 36–42.
[4] E. S. Ochotta, T. Mukherjee, R. A. Rutenbar, and L. R. Carley, *Practical Synthesis of High Performance Analog Circuits*. Norwell, MA: Kluwer, 1999.
[5] T. Massier, H. Graeb, and U. Schlichtmann, "The sizing rules method for CMOS and bipolar analog integrated circuit synthesis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 12, pp. 2209–2222, Dec. 2008.
[6] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, and J. R. Hellums, "ANACONDA: Simulation-based synthesis of analog circuits via stochastic pattern search," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 6, pp. 703–717, Jun. 2000.
[7] H. E. Graeb, *Analog Design Centering and Sizing*. New York: Springer-Verlag, 2007.
[8] Y. Xu, K.-L. Hsiung, X. Li, I. Nausieda, S. Boyd, and L. Pileggi, "OPERA: Optimization with ellipsoidal uncertainty for robust analog IC design," in *Proc. Des. Autom. Conf.*, 2005, pp. 632–637.
[9] B. De Smedt and G. G. E. Gielen, "HOLMES: Capturing the yield-optimized design space boundaries of analog and RF integrated circuits," in *Proc. Des. Autom. Test Eur. Conf.*, 2003, pp. 256–261.

[10] S. K. Tiwary, P. K. Tiwary, and R. A. Rutenbar, "Generation of yield-aware Pareto surfaces for hierarchical circuit design space exploration," in *Proc. Des. Autom. Conf.*, 2006, pp. 31–36.
[11] X. Li, P. Gopalakrishnan, Y. Xu, and L. Pileggi, "Robust analog/RF circuit design with projection-based performance modeling," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 1, pp. 2–15, Jan. 2007.
[12] G. Yu and P. Li, "Yield-aware analog integrated circuit optimization using geostatistics motivated performance modeling," in *Proc. Int. Conf. Comput. Aided Des.*, 2007, pp. 464–469.
[13] P. Drennan and C. McAndrew, "A comprehensive MOSFET mismatch model," in *IEDM Tech. Dig.*, 1999, pp. 167–170.
[14] J. Nocedal and S. Wright, *Numerical Optimization*. New York: Springer-Verlag, 1999.
[15] J. H. Friedman, "Stochastic gradient boosting," *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, Feb. 2002.
[16] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[17] NIST, *NIST/SEMATECH e-Handbook of Statistical Methods*, Gaithersburg, MD, 2003, last accessed Jul. 18, 2006. [Online]. Available: http://www.itl.nist.gov/div898/handbook/pmc/section1/pmc16.htm
[18] G. van der Plas, G. G. E. Gielen, and W. M. C. Sansen, *A Computer-Aided Design and Synthesis Environment for Analog Integrated Circuits*. New York: Springer-Verlag, 2002.
[19] T. McConaghy and G. G. E. Gielen, "Analysis of simulation-driven numerical performance modeling techniques for application to analog circuit optimization," in *Proc. ISCAS*, May 2005, pp. 1298–1301.
[20] G. G. E. Gielen, "Techniques and applications of symbolic analysis for analog integrated circuits: A tutorial overview," in *Computer Aided Design of Analog Integrated Circuits and Systems*, R. A. Rutenbar, G. G. E. Gielen, and B. A. A. Antao, Eds. Piscataway, NJ: IEEE Press, 2002, pp. 245–261.
[21] W. Dames, B. De Smedt, E. Lauwers, E. Yannis, and W. Verhaegen, "Method and apparatus for designing electronic circuits," US Number US20050257178, Nov. 17, 2005.
[22] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Glob. Optim.*, vol. 13, no. 4, pp. 455–492, Dec. 1998.
[23] F. Schenkel, M. Pronath, S. Zizala, R. Schwencker, H. Graeb, and K. Antreich, "Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search," in *Proc. Des. Autom. Conf.*, 2001, pp. 858–863.
[24] G. Stehr, H. Graeb, and K. Antreich, "Analog performance space exploration by normal-boundary intersection and Fourier–Motzkin elimination," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 10, pp. 1733–1748, Oct. 2007.
[25] K. Antreich, J. Eckmueller, H. E. Graeb, M. Pronath, F. Schenkel, R. Schwencker, and S. Zizala, "WiCkeD: Analog circuit synthesis incorporating mismatch," in *Proc. Custom Integr. Circuits Conf.*, 2000, pp. 511–514.
[26] Y. Xu, L. T. Pileggi, and S. P. Boyd, "ORACLE: Optimization with recourse of analog circuits including layout extraction," in *Proc. Des. Autom. Conf.*, 2004, pp. 151–154.
[27] X. Li, J. Le, P. Gopalakrishnan, and L. Pileggi, "Asymptotic probability extraction for nonnormal performance distributions," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 1, pp. 16–37, Jan. 2007.
[28] M. D. McKay, W. J. Conover, and R. J. Beckman, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, May 1979.
[29] M. J. Sasena, "Flexibility and efficiency enhancements for constrained global design optimization with Kriging approximations," Ph.D. dissertation, Univ. Michigan Press, Ann Arbor, MI, 2002.
[30] N. Ampazis and S. J. Perantonis, "Two highly efficient second order algorithms for training feedforward networks," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1064–1074, Sep. 2002.
[31] M. J. D. Powell, "The NEWUOA software for unconstrained optimization without derivatives," in *Large Scale Nonlinear Optimization*, G. Di Pillo and M. Roma, Eds. The Netherlands: Springer-Verlag, 2006, pp. 255–297.
[32] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001.
[33] G. S. Hornby, "ALPS: The age-layered population structure for reducing the problem of premature convergence," in *Proc. GECCO*, M. Keijzer, Ed., 2006, vol. 1, pp. 815–822.
[34] D. Yuret, "From genetic algorithms to efficient optimization," M.S. thesis, MIT AI Lab., Cambridge, MA, 1994.
[35] T. G. Kolda, R. M. Lewis, and V. Torczon, "Optimization by direct search: New perspectives on some classical and modern methods," *SIAM Rev.*, vol. 45, no. 3, pp. 385–482, 2003.
[36] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, pp. 212–229, Apr. 1961.
[37] N. Jardine and R. Sibson, "The construction of hierarchic and non-hierarchic classifications," *Comput. J.*, vol. 11, no. 2, pp. 177–184, 1968.
[38] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," *J. Amer. Stat. Assoc.*, vol. 22, pp. 209–212, 1927. [Online]. Available: http://en.wikipedia.org/wiki/Binomial_proportion_confidence_interval

Trent McConaghy (S'95–M'99) received the B.E. degree (with great distinction) in engineering and the B.S. degree (with great distinction) in computer science from the University of Saskatchewan, Saskatoon, SK, Canada, in 1999 and the Ph.D. degree in electrical engineering from Katholieke Universiteit Leuven, Leuven, Belgium, in 2008.

He was a cofounder and Chief Scientist with Analog Design Automation, Inc., which was acquired by Synopsys, Inc., in 2004. Prior to that, he did research for the Canadian Department of National Defense. He is cofounder and Chief Scientific Officer with Solido Design Automation, Inc., Saskatoon. He has about 40 peer-reviewed technical papers and patents granted/pending. He has given invited talks/tutorials at many laboratories, universities, and conferences such as Jet Propulsion Laboratories, Massachusetts Institute of Technology, International Conference on Computer-Aided Design (CAD), and Design Automation Conference. He is regularly a technical program committee member and reviewer in both the CAD and intelligent systems fields, such as IEEE TRANSACTIONS ON CAD, *Association for Computing Machinery Transactions on Design Automation of Electronic Systems, Electronics Letters*, to IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the *Journal of Genetic Programming and Evolvable Machines*, Genetic Programming Theory and Practice, Genetic and Evolutionary Computation Conference, International Conference on Evolvable Systems, etc. His research interest is in statistical machine learning and intelligent systems, with transistor-level CAD applications such as variation-aware design, analog topology design, automated sizing, knowledge extraction, and symbolic modeling.

Georges G. E. Gielen (S'87–M'92–SM'99–F'02) received the M.Sc. and Ph.D. degrees in electrical engineering from Katholieke Universiteit Leuven, Leuven, Belgium, in 1986 and 1990, respectively.

He is currently a Full Professor with the Department of Electrotechnical Engineering—Microelectronics and Sensors (ESAT—MICAS), Katholieke Universiteit Leuven. His research interests are in the design of analog and mixed-signal integrated circuits, and particularly in analog and mixed-signal computer-aided design (CAD) tools and design automation (modeling, simulation and symbolic analysis, analog synthesis, analog layout generation, analog and mixed-signal testing). He is coordinator or partner of several (industrial) research projects in this area, including several European projects (European Union, Microelectronics Development for European Applications, European Space Agency). He has authored or coauthored five books and more than 300 papers in edited books, international journals, and conference proceedings. He regularly is a member of the Program Committees of international conferences (Design Automation Conference, International Conference on CAD, International Symposium on Circuits and Systems (CAS), Design, Automation and Test in Europe (DATE), Custom Integrated Circuits Conference, . . .), and served as General Chair of the DATE conference in 2006 and of the International Conference on CAD in 2007. He serves regularly as member of editorial boards of international journals (IEEE TRANSACTIONS ON CAS, *Springer International Journal on Analog Integrated Circuits and Signal Processing, Elsevier Integration*).

Dr. Gielen received the 1995 Best Paper Award in the John Wiley international journal on Circuit Theory and Applications, and was the 1997 Laureate of the Belgian Royal Academy on Sciences, Literature and Arts in the discipline of Engineering. He received the 2000 Alcatel Award from the Belgian National Fund of Scientific Research for his innovative research in telecommunications, and won the DATE 2004 Best Paper Award. He served as an elected member of the Board of Governors of the IEEE CAS Society and as Chairman of the IEEE Benelux CAS chapter. He served as the President of the IEEE CAS Society in 2005. He was elected DATE Fellow in 2007, and received the IEEE Computer Society Outstanding Contribution Award and the IEEE CAS Society Meritorious Service Award in 2007.