Convexity-based Algorithms for Design Centering

Sachin S. Sapatnekar, Pravin M. Vaidya and Sung-Mo Kang

Abstract

A new technique for design centering, and for polytope approximation of the feasible region for a design are presented. In the first phase, the feasible region is approximated by a convex polytope, using a method based on a theorem on convex sets. As a natural consequence of this approach, a good approximation to the design center is obtained. In the next phase, the exact design center is estimated using one of two techniques that we present in this paper. The first inscribes the largest Hessian ellipsoid, which is known to be a good approximation to the shape of the polytope, within the polytope. This represents an improvement over previous methods, such as simplicial approximation, where a hypersphere or a crudely estimated ellipsoid is inscribed within the approximating polytope. However, when the pdf's of the design parameters are known, the design center does not necessarily correspond to the center of the largest inscribed ellipsoid. Hence, a second technique is developed, which incorporates the probability distributions of the parameters, under the assumption that their variation is modeled by Gaussian probability distributions. The problem is formulated as a convex programming problem and an efficient algorithm is used to calculate the design center, using fast and efficient Monte Carlo methods to estimate the yield gradient. An example is provided to illustrate how ellipsoid-based methods fail to incorporate the probability density functions, and is solved using the convex programming-based algorithm.

Convexity-based Algorithms for Design Centering

Sachin S. Sapatnekar, Pravin M. Vaidya and Sung-Mo Kang

I. INTRODUCTION

While manufacturing a circuit, it is inevitable that process variations will cause design parameters, such as component values, to waver from their nominal values. As a result, the manufactured circuit may no longer meet some behavioral specifications, such as requirements on the delay, gain and bandwidth, that it has been designed to satisfy. The procedure of design centering attempts to select the nominal values of design parameters so as to ensure that the behavior of the circuit remains within specifications, with the greatest probability. In other words, the aim of design centering is to ensure that the manufacturing yield is maximized. Previous approaches to solving this problem have traditionally taken two routes :

(i) The *statistical* approach

These are primarily Monte Carlo techniques which utilize information gathered from simulating the circuit behavior for various sets of design parameters. Monte Carlo-based methods have the disadvantage of being computationally expensive when the targeted yield is high [1]; however, in the limiting case where every point in the space is sampled, they give perfect accuracy.

(ii) The *geometrical* approach

The feasible region in the space of design parameters, i.e., the region where the behavioral specifications are satisfied, is approximated by a known geometrical body, such as a polytope or an ellipsoid. The center of this body is then approximated, and is taken to be the design center. Such approaches frequently assume that the feasible region is convex and bounded. These methods commonly suffer from the following drawbacks:

(a) Limitations associated with the types of geometric bodies that are typically used to approximate the feasible region:

- In the case of *ellipsoidal* approximation as in [2], certain convex bodies cannot be approximated accurately. This is because an ellipsoid is symmetric about any hyperplane passing through its

center, and is inherently incapable of producing a good approximation to a body that has a less symmetric structure.

- A *polytope* can provide a better approximation to a convex body than an ellipsoid, regardless of how symmetric the convex body is, since any convex body can be thought of as a polytope with an infinite number of faces. However, finding the center of a polytope is computationally complex [3] and cannot be carried out in a reasonable time.

  The simplicial approximation algorithm [4] attempts to inscribe the largest hypersphere in the polytope, and takes its center as the design center; however, as pointed out in [4] itself, in the case of elongated bodies, such as a rectangle with a highly skewed aspect ratio, it would be more appropriate to inscribe a ellipsoid rather than a hypersphere. A crude and approximate method for this purpose is outlined in [4]. In any case, the simplicial approximation procedure essentially amounts to approximating the feasible region by a polytope, and then approximating the polytope by a hypersphere or ellipsoid. Hence, it suffers from the drawbacks of ellipsoidal approximation listed above.

(b) As pointed out above, the methods in [2, 4] essentially approximate the feasible region by means of an ellipsoid, and take the center of that ellipsoid to be the design center, regardless of the probability distributions that define variations in the design parameters. However, as illustrated in the example in Section IV B, the design center could be highly dependent on the exact probability distributions of the variables, and would change according to these distributions.

(c) Real feasible regions are seldom convex. While in many cases, they are "nearly convex," there are documented cases where the feasible region is not very well-behaved. In a large number of cases of good designs, since the joint probability density function of the statistical variables decays quite rapidly from the design center, a convex approximation does not adversely affect the result. However, if the nominal design has a very poor yield, a convex approximation will prove to be inadequate.

Recently, a third class of approaches has been proposed which hybridizes the statistical and geometric approaches [5, 6]; this class is less mature than the statistical and geometrical approaches. These approaches build up statistical models to approximate the behavioral functions of the circuit, and to use these models to find the optimal values of the nominal parameters. These methods trade off the accuracy of

statistical modeling with the cost of modeling and the ease with which one may work with the approximating functions in finding the optimum.

However, when all is said and done, each class of approaches has its own advantages and disadvantages, and it is not possible to predict which one of the above classes would provide a better result than another. In a real-life situation, it would be advisable for a designer to find solutions provided by all the available CAD tools, and to then select the best solution.

In this work, we explore the route of geometrical approaches to solving the design centering problem and introduce a new convexity-based design centering approach. In this approach, the feasible region is first approximated by a convex polytope. As a natural consequence of this procedure, we can obtain a reasonably good approximation to the actual design center. In the next phase, the design center is found using one of two new *geometrical* approaches.

(I) The first approach, called the largest Hessian ellipsoid (L.H.E.) method, inscribes the largest Hessian ellipsoid, defined in Section III B, inside the polytope. The shape of the Hessian ellipsoid is well-known to be representative of the shape of the polytope [3]. This method presents an improvement over [4], in which the center of either a hypersphere, or a crudely chosen ellipsoid, is taken to be the design center. This approach is primarily directed towards problems where one has no knowledge of the statistical variation of the design parameters, and would like to find a design center that is "deep within" the feasible region, but as shown by experimental results, also gives satisfactory results in general.

(II) The second approach, the convex programming (C.P.) approach, proceeds by formulating the design centering problem as a convex programming problem, assuming that the variations in the design parameters are modeled by Gaussian probability distributions. A convex programming algorithm [3], whose efficacy has been illustrated in [7,8], is used to find the solution to this problem. This method differs from other geometrical methods, such as [2,4], in one important respect: information about the probability distribution is explicitly utilized, so that, in general, a different design center is found when the covariance matrix for the random variables is changed.

The suggested approaches are applicable not only to circuit design, but to other manufacturing processes wherein the feasible region can be approximated by a convex body, and in the case of the second approach, the variation of the nominal design variables can be described by a Gaussian distribution.

The remainder of this paper is organized as follows. Section II describes our algorithm for approximating the feasible region by a convex polytope. The two approaches for finding the design center are described in detail in Section III. Experimental results are provided in Section IV, followed by concluding remarks in Section V.

II. APPROXIMATION OF THE FEASIBLE REGION

A. Introduction

The feasible region $\mathcal{F} \subset \mathbf{R}^n$, where n is the number of design parameters, is defined as the set of points in the design parameter space for which the circuit satisfies all specifications on its behavior. A common assumption made by design centering algorithms is that $\mathcal{F}$ is a convex body. In this work, we preserve that assumption, and use properties of convex sets to create an approximation to $\mathcal{F}$.

Previously, to find a polytope approximation to the feasible region, the simplicial approximation method [4] was used. This method proceeds as follows:

(a) Determine a set of $m \geq n+1$ points on the boundary of $\mathcal{F}$.

(b) Find the convex hull of these points and use this polyhedron as the initial approximation to $\mathcal{F}$. Set $k = 0$.

(c) Inscribe the largest n-dimensional hypersphere in this approximating polyhedron and take its center as the first estimate of the design center. This process involves the solution of a linear program.

(d) Find the midpoint of the largest face of the polyhedron, i.e., the face in which the largest $(n-1)$-dimensional hypersphere can be inscribed.

(e) Find a new boundary point on $\mathcal{F}$ by searching along the outward normal of the largest face found in Step (d) extending from the midpoint of this face. This is carried out by performing a line search.

(f) Inflate the polyhedron by forming the convex hull of all previous points, plus the new point generated in Step (e).

(g) Find the center of the largest hypersphere inscribed in the new polyhedron found in Step (f). This involves the solution of a linear program. Set $k = k+1$, and go to Step (d).

Thus, each iteration involves the solution of about $n+1$ linear programs [9] a line search, and updating a convex hull. The number of linear programs to be solved in the entire procedure is large (specifically, $k(n+1)$, where k is the number of iterations). Moreover, the procedure of updating the convex hull is computationally expensive since updating a convex hull consists of identifying hyperplanes to be removed and finding the equations of the new hyperplanes. Obtaining the equation of a hyperplane, given the vertices, is an $O(n^3)$ operation; this needs to be carried out for each new hyperplane of the polytope while updating the convex hull. At the end of this procedure, the approximation to the design center is available.

We compare the complexity of simplicial approximation with that of Algorithm I, outlined in Section III B, whose strategy, like simplicial approximation, is to approximate the feasible region by a polytope and inscribe the largest ellipsoid in the polytope. Generation of each new hyperplane of the polytope involves a line search. After up to $2n$ line searches, a new polytope center has to be computed, which is an $O(n^{2.5})$ operation. Finally, after the polytope has been approximated, a small number of linear programs have to be solved to find the design center. Thus, the algorithm that we present is computationally less complex than simplicial approximation, and is more accurate for convex feasible regions, since a more representative ellipsoid is used for inscription in the polytope.

In practise, however, the simulations take times that are at least an order of magnitude greater than that taken for the solution of linear programs and other updates, and hence, the improvement in computational efficiency here may not be very noticeable.

Algorithm II, presented in Section III C, is an improvement over both simplicial approximation and Algorithm I for convex feasible regions, since it takes into account the dependence of the design center on the probability distributions of the design parameters. Moreover, for convex feasible regions, this approach are an improvement over that presented in [2], where a symmetric body, an ellipsoid, is used to approximate the feasible region. This is because a polytope approximation method would show no loss in the quality of the solution, even when the feasible region has diminished symmetry properties in relation to ellipsoids.

B. Construction of the Approximating Polytope

The Supporting Hyperplane Theorem

Our algorithm is based on the following well-known theorem on convex sets [10] that is stated below.

Theorem : Let C be a convex set and let $\mathbf{y}$ be a boundary point of C. Then there exists a hyperplane containing $\mathbf{y}$ and containing C in one of its closed half-spaces.

Definition : A tangent plane at a point $\mathbf{z}_0$ on the boundary of a region is given by

$$[\nabla g(\mathbf{z}_0)]^T \mathbf{z} = [\nabla g(\mathbf{z}_0)]^T \mathbf{z}_0 \tag{1}$$

where $g(\mathbf{z}) \geq 0$ is an active constraint at point $\mathbf{z}_0$, and $\nabla g(.)$ is the gradient of the function $g(.)$.

For a point $\mathbf{z}_0$ on the boundary of a convex set, C, a tangent plane at $\mathbf{z}_0$ satisfies the supporting hyperplane property. In particular, for the constraint

$$g(\mathbf{z}) \geq 0, \tag{2}$$

C is contained in the half-space

$$[\nabla g(\mathbf{z}_0)]^T \mathbf{z} \geq [\nabla g(\mathbf{z}_0)]^T \mathbf{z}_0 . \tag{3}$$

Outline of the Algorithm

The aim of the algorithm is to approximate the feasible region, $\mathcal{F} \subset \mathbf{R}^n$ by a polytope

$$\mathcal{P} = \{\mathbf{x} \mid A\mathbf{z} \geq \mathbf{b}\}, A \in \mathbf{R}^{m \times n}, \mathbf{b} \in \mathbf{R}^m, \tag{4}$$

formed by the intersection of m half-spaces in $\mathbf{R}^n$.

The algorithm begins with an initial feasible point, $\mathbf{z}_0 \in \mathcal{F} \subset \mathbf{R}^n$. An n-dimensional box, namely, $\{\mathbf{z} \in \mathbf{R}^n \mid z_{min} \leq z_i \leq z_{max}\}$, containing $\mathcal{F}$ is chosen as the initial polytope $\mathcal{P}_0$. In each iteration, n orthogonal search directions, $\mathbf{d}_1, \mathbf{d}_2 \cdots \mathbf{d}_n$ are chosen. In our experiments, these were taken to be the n coordinate directions. A binary search is conducted from $\mathbf{z}_0$ to identify a boundary point $\mathbf{z}_{\mathbf{b}_i}$ of $\mathcal{F}$, for each direction $\mathbf{d}_i$. If $\mathbf{z}_{\mathbf{b}_i}$ is relatively deep in the interior of $\mathcal{P}$, then the tangent plane to $\mathcal{F}$ at $\mathbf{z}_{\mathbf{b}_i}$ is added to the set of constraining hyperplanes in Equation (4). A similar procedure is carried out along the direction $-\mathbf{d}_i$. Once all of the hyperplanes have been generated, the center of the new polytope is calculated, using a method described in the next subsection. Then $\mathbf{z}_0$ is reset to be this center, and the above process is repeated.

The algorithm is described by the following piece of pseudo-code.

```
k = 0
Initialize P_0 and z_0
While (Number of planes added in the last iteration ≠ 0) {
        P_{k+1} = P_k
        Choose orthogonal directions d_1, d_2 ··· d_n
        For i = 1 to n {
                Perform a binary search along direction d_i from z_0
                                to find a boundary point z_{b_i} of F
                If z_{b_i} exists, z_{b_i} ∈ P_{k+1}, and distance(z_{b_i}, boundary of P_{k+1}) > ε {
                   Add a hyperplane to the polytope
                   P_{k+1} = P_{k+1} ∪ new hyperplane
                }
                Perform a binary search along direction −d_i from z_0
                                to find a boundary point z_{b_i} of F
                If z_{b_i} exists, z_{b_i} ∈ P_{k+1}, and distance(z_{b_i}, boundary of P_{k+1}) > ε {
                   Add a hyperplane to the polytope
                   P_{k+1} = P_{k+1} ∪ new hyperplane
                }
        }
        Set z_0 = center of the updated polytope, P_{k+1}
        k = k + 1;
}
```

For the algorithm to provide a good approximation, the thickness of the polytope in each dimension should be of similar orders of magnitude; if not, the algorithm is liable to terminate too early. This may be achieved by getting the designers input on the approximate order of magnitude tolerances for each parameter, and appropriately scaling the axes so that all tolerances are of a similar order of magnitude.

The procedure requires a technique for calculating the gradient of an active constraint at a boundary point. As in [2], one of several methods may be employed for this purpose. If the exact functional form of the constraint is known, an expression for the gradient may be derived. If not, methods such as the adjoint network technique [11] or finite differences may be used to calculate the gradient.

An example of the polytope approximation procedure is illustrated in Section IV A.

C. Computation of the Center of the Polytope

Since finding the exact volumetric center of a polytope is computationally difficult [3] and is not essential to our algorithm, we settle for finding an approximation to the center, i.e., a point that is deep within the interior of the polytope, and can be found through relatively inexpensive computation.

Consider a polytope P defined by (4), and let $\mathbf{a_i}^T$ be the i^{th} row of matrix $A \in \mathbf{R}^{m \times n}$, and b_i be the i^{th} element of $\mathbf{b} \in \mathbf{R}^m$. The center $\mathbf{z}_c$, is taken to be the point that minimizes the following *log-barrier*

function

$$F(\mathbf{z}) = -\sum_{i=1}^{m} \log_e(\mathbf{a_i}^T \mathbf{z} - b_i). \tag{5}$$

Note that near the boundary of the polytope, $F(\mathbf{z})$ tends to infinity and its value decreases as one moves *deeper* into the interior of the polytope. Also, the value of $F(\mathbf{z})$ is undefined outside the boundary of the polytope. Moreover, F is a convex function of $\mathbf{z} \in P$, with a $1 \times n$ gradient vector

$$\nabla F(\mathbf{z}) = -\sum_{i=1}^{m} \frac{\mathbf{a_i}^T}{(\mathbf{a_i}^T \mathbf{z} - b_i)} \tag{6}$$

and an $n \times n$ Hessian matrix

$$\mathcal{H}(\mathbf{z}) = \nabla^2 F(\mathbf{z}) = \sum_{i=1}^{m} \frac{\mathbf{a_i} \mathbf{a_i}^T}{(\mathbf{a_i}^T \mathbf{z} - b_i)^2}. \tag{7}$$

Since the initial polytope is a box, its center is easy to find. At each subsequent iteration, hyperplanes are added to the polytope. The new center is found iteratively using a modified Newton's method [10]. The initial point $\mathbf{z}_0$ for the modified Newton's method is found by moving halfway to the closest boundary in the direction normal to the newly added plane. The point $\mathbf{z}_0$ thus obtained is guaranteed to be in the interior of the new polytope.

The modified Newton method for finding the center $\mathbf{z}_c$ then generates iterates of the form

$$\mathbf{z}_{k+1} = \mathbf{z}_k + t^* \xi_k \tag{8}$$

for $k = 0, 1, 2, \cdots$, until convergence, where ξ_k is the *Newton direction* at $\mathbf{z}_k$ given by

$$\xi_k = -\mathcal{H}^{-1}(\mathbf{z}_k)[\nabla F(\mathbf{z}_k)]^T = -[\nabla^2 F(\mathbf{z}_k)]^{-1}[\nabla F(\mathbf{z}_k)]^T \tag{9}$$

and t^* is the point that minimizes the one-dimensional function

$$\phi(t) = F(\mathbf{z}_k + t\xi_k). \tag{10}$$

t^* may be obtained by performing a simple one-dimensional line-search.

Note that the process of computing a Newton direction by (9) involves the inversion of an $n \times n$ Hessian matrix which takes $O(n^3)$ time and can prove to be rather expensive. This expense can be cut down by maintaining the inverse of an approximate Hessian $\hat{\mathcal{H}}$ via rank-one updates [10], and using an

approximate Newton direction $\hat{\xi}_k$ instead of ξ_k in the line search. We note that using an approximate Newton direction instead of the exact one essentially does not affect the convergence properties of the center-finding algorithm [3]. Details of computation of the approximate inverse Hessian are provided in [7].

III. FINDING THE DESIGN CENTER

A. Definition of the Design Center

The random variations in the values of the design parameters are modeled by a probability density function, $\Phi(\mathbf{z}) : \mathbf{R}^n \rightarrow [0, 1]$, with a mean corresponding to the nominal value of the design parameters. $\Phi(\mathbf{z})$ is typically taken to be a multivariate Gaussian density function. The yield of the circuit, Y, as a function of the mean, $\mathbf{x}$, is given by

$$\mathrm{Y}(\mathbf{x}) = \int_{\mathcal{F}} \Phi_{\mathbf{X}}(\mathbf{z}) d\mathbf{z} \tag{11}$$

where $\mathcal{F}$ represents the feasible region, where the design parameters are such that the circuit satisfies its behavioral requirements.

The *design center* is the point $\mathbf{x}$ at which the yield, $\mathrm{Y}(\mathbf{x})$, is maximized.

In the particular case of an ellipsoidal feasible region and a multivariate Gaussian density function, it can be shown that the design center corresponds to the center of the ellipsoid.

Based on this fact, the simplicial approximation method [4] attempts to inscribe the largest hypersphere within the approximating polytope. This procedure involves the solution of a linear program. Note that the procedure, in essence, approximates the polytope by a sphere; hence, the solution to the design centering problem is the center of the approximating sphere. A disadvantage of this method is that it would not give the best results for elongated regions of acceptability, such as rectangles. A more realistic center would be obtained by inscribing the largest ellipsoid inside the polytope. In [4], a simple estimate of the spread in each design parameter is used to define the shape of the largest ellipsoid that can be inscribed within the polytope. The largest ellipsoid of that shape is then inscribed into the polytope (it may be pointed out that this ellipsoid is not necessarily the largest of all ellipsoids that can be inscribed within the polytope). This method is clearly very approximate since only a limited set of ellipsoids, whose major axes are along the coordinate directions, is considered. Hence, an ellipsoidal shape defined by parameter spreads does not necessarily correspond to the largest ellipsoid that can be inscribed within the polytope. A better

method of determining the shape of the largest inscribed ellipsoid is required; in this paper, we present an approach for doing so.

B. Algorithm I : Inscribing the Largest Hessian ellipsoid

Let E($\mathbf{x}$,$\mathcal{B}$,r) denote the ellipsoid

$$\left\{\mathbf{y} \mid (\mathbf{y}-\mathbf{x})^T \mathcal{B} (\mathbf{y}-\mathbf{x}) \leq \mathrm{r}^2\right\}. \tag{12}$$

The Hessian ellipsoid at a point $\mathbf{x}$ in the polytope P, namely the ellipsoid E($\mathbf{x}$,$\mathcal{H}(\mathbf{x})$,1), where $\mathcal{H}(\mathbf{x}_c)$ is the Hessian of the log-barrier function (Equation (5)) is a good approximation to the polytope locally around $\mathbf{x}$ [3]. Hence, the goal is to find the largest ellipsoid in the class E($\mathbf{x}$, $\mathcal{H}(\mathbf{x})$,r) that can be inscribed in the polytope, and its center $\mathbf{x}_c$. The point $\mathbf{x}_c$ will be taken to be the computed design center.

The minimum of the log-barrier function gives a good approximation to $\mathbf{x}_c$. Therefore, it may be used as an initial guess for the iterative process, described below, that is carried out to find $\mathbf{x}_c$.

distance $= \infty$
$\mathbf{x}_c$ = *Minimizer of the log barrier function (Equation (5))*
while (distance $> \epsilon$*)* {
    $\mathbf{x}_{old} = \mathbf{x}_c$
    $\mathcal{H}$ = *Hessian at* $\mathbf{x}_c$
    Inscribe the largest ellipsoid of the type E($\mathbf{x}$, $\mathcal{H}(\mathbf{x}_c)$, r*)*
        (Note that the ellipsoid shape, determined by $\mathcal{H}(\mathbf{x}_c)$*,*
        is fixed, and that $\mathbf{x}$ *and* r *are allowed to vary)*
    Set $\mathbf{x}_c$ = *center of this ellipsoid*
    distance $= \|\mathbf{x}_c - \mathbf{x}_{old}\|$
}

In each iteration, the shape of the ellipsoid is fixed by the Hessian, $\mathcal{H}$, computed at the current value of $\mathbf{x}_c$; $\mathbf{x}$ and r are allowed to vary. It must be pointed out here that the Hessian matrix $\mathcal{H}$, is positive definite.

The process of inscribing the largest hypersphere is described in [4]. For the case of a hyperellipsoid, E($\mathbf{x}$,$\mathcal{H}$,r), where $\mathcal{H}$ is positive definite, a linear transformation is first performed. Since $\mathcal{H}$ is positive definite, it can be written in terms of its Cholesky factors, i.e., $\mathcal{H} = \mathcal{H}_c^T \mathcal{H}_c$. Hence, the ellipsoid E($\mathbf{x}$,$\mathcal{H}$,r) can be written as

$$(\mathbf{y}-\mathbf{x})^T \mathcal{H}_c^T \mathcal{H}_c (\mathbf{y}-\mathbf{x}) \leq \mathrm{r}^2. \tag{13}$$

The transformation, $\mathbf{w} = \mathcal{H}_c(\mathbf{y}-\mathbf{x})$, maps the above equation into

$$\mathbf{w}^T \mathbf{w} \leq \mathrm{r}^2, \tag{14}$$

which is the equation of a hypersphere. The solution of this new problem of inscribing the largest hypersphere in a transformed polytope involves the solution of a linear program, as described in [4].

C. Algorithm II : The Convex Programming Approach

Formulation of the convex program

When the probability density functions that represent variations in the design parameters are Gaussian in nature, the design centering problem can be posed as a convex programming problem.

The joint Gaussian probability density function of n independent random variables $\mathbf{z} = (z_1, \cdots, z_n)$, where z_i has mean x_i and variance σ_i, is given by

$$\Phi_{\mathbf{X}}(\mathbf{z}) = \frac{1}{(2\pi)^{n/2}\sigma_1\sigma_2\cdots\sigma_n} \exp\left[\sum_{i=0}^{i=n} -\frac{(z_i - x_i)^2}{2\sigma_i^2}\right] \tag{15}$$

where $\mathbf{x} = (x_1, \cdots, x_n)$.

This is known to be a log-concave function of $\mathbf{x}$ and $\mathbf{z}$. Also, note that arbitrary covariance matrices can be handled, since a symmetric matrix may be converted into a diagonal form by a simple linear (orthogonal) transformation.

The design centering problem is now formulated as

$$\begin{aligned} \text{maximize} \quad & \mathrm{Y}(\mathbf{x}) = \textstyle\int_{\mathcal{P}} \Phi_{\mathbf{X}}(\mathbf{z}) d\mathbf{z} \\ \text{such that} \quad & \mathbf{x} \in \mathcal{P}. \end{aligned} \tag{16}$$

where $\mathcal{P}$ is the polytope approximation to the feasible region $\mathcal{F}$, found in Section II. It is a known fact that the integral of a log-concave function over a convex region is also a log-concave function [12]. Thus, the yield function $\mathrm{Y}(\mathbf{x})$ is log-concave, and the above problem reduces to a problem of maximizing a log-concave function over a convex set. Hence, this can be transformed into a convex programming problem, with the corresponding property that any local minimum of the problem is a global minimum. It is worth noting here that the yield function remains convex as long as $\phi_{\mathbf{X}}(\mathbf{z})$ is any log-concave function of $\mathbf{x}$ and $\mathbf{z}$. For example, this approach would also be valid for an exponential probability density function.

Applying the Convex Programming Algorithm

The convex programming algorithm proposed in [3] provides an efficient technique for solving a convex programming problem, such as (16). We define the *feasible set*

$$S = \{\mathbf{x} \in \mathbf{R}^n \mid \mathbf{x} \in \mathcal{P}\} \tag{17}$$

and let $\mathbf{x}_c$ be the solution to (16). Initially, a polytope $\mathcal{Q} = \mathcal{P}$ that contains $\mathbf{x}_c$ is chosen. The polytope $\mathcal{Q}$ is given by

$$\mathcal{Q} = \{\mathbf{x} \mid \hat{A}\mathbf{z} \geq \hat{\mathbf{b}}\}, \hat{A} \in \mathbf{R}^{p \times n}, \hat{b} \in \mathbf{R}^n. \tag{18}$$

The algorithm proceeds iteratively as follows. First, a *center* $\mathbf{z}_c$, deep in the interior of the current polytope $\mathcal{Q}$ is found, by minimizing the log-barrier function

$$F(\mathbf{z}) = -\sum_{i=1}^{p} \log_e(\hat{\mathbf{a}}_\mathbf{i}^T \mathbf{z} - \hat{b}_i) \tag{19}$$

where $\hat{\mathbf{a}}_\mathbf{i}^T$ is the i^{th} row of matrix $\hat{A}$ and $\hat{b}_i$ be the i^{th} element of $\hat{\mathbf{b}}$. The minimization procedure is as given in Section II C.

There exists a hyperplane that divides the polytope into two parts, such that $\mathbf{x}_c$ is contained in one of them, satisfying the constraint

$$\mathbf{c}^T \mathbf{z} \geq \mathbf{c}^T \mathbf{z}_c \tag{20}$$

$$\text{with } \mathbf{c} = -[\nabla \mathrm{Y}(\mathbf{x})]^T \tag{21}$$

being the negative of the gradient of the yield (objective) function,

Since the yield function is not available in an explicit form, the gradient is estimated using the yield gradient approximation method presented in Section III D. This yield estimator works with the polytope approximation of the feasible region and requires no simulations. A point is considered to be feasible if it lies within the approximating polytope; this leads to a substantial savings in computation, since it is much cheaper to find out whether a point lies within a polytope than to simulate the circuit with a new set of parameter values.

In practice, the yield gradient is approximate, and possibly erroneous, as it is based on Monte Carlo simulation. To offset this problem, the new hyperplane is taken as

$$\mathbf{c}^T \mathbf{z} \geq \mathbf{c}^T \mathbf{z}_c - \delta \mid \mathbf{c}^T \mathbf{z}_c \mid \tag{22}$$

where δ is a small positive number (typically 0.1 or 0.2), representing the fact that the plane is moved away by a certain fraction towards the boundary of the current polytope. Qualitatively speaking, the hyperplane given by Equation (22) shaves off less from the polytope given by Equation (20), thereby reducing the possibility of errors due to incorrect gradient estimations.

The constraint in Equation (22) is added to the current polytope to give a new polytope, $\mathcal{Q}$, that has roughly half the original volume. The process is repeated until the polytope is sufficiently small, and the final center $\mathbf{z}_c$ is taken to be the computed design center.

D. Yield Gradient Estimation

The yield function, defined in Equation (11), can also be written as

$$\mathrm{Y}(\mathbf{x}) = \int \cdots \int_{\mathcal{F}} h(\mathbf{z}) \Phi_{\mathbf{X}}(\mathbf{z}) d\mathbf{z} \tag{23}$$

where $h(\mathbf{z}) = 0$ if $\mathbf{z} \notin \mathcal{F}$, and $h(\mathbf{z}) = 1$ if $\mathbf{z} \in \mathcal{F}$.

In our method, since we have a polytope approximation, $\mathcal{P}$, to the feasible region, $\mathcal{F}$, we can take an approximation to the yield as

$$\mathrm{Y}_{\mathrm{approx}}(\mathbf{x}) = \int \cdots \int_{\mathcal{P}} g(\mathbf{z}) \Phi_{\mathbf{X}}(\mathbf{z}) d\mathbf{z} \tag{24}$$

where $g(\mathbf{z}) = 0$ if $\mathbf{z} \notin \mathcal{P}$, and $g(\mathbf{z}) = 1$ if $\mathbf{z} \in \mathcal{P}$. Thus, the computation of $g(\mathbf{z}_k)$ is simply a matter of checking whether the point lies within the polytope or not, which is a computationally cheap operation and *does not* require an actual circuit simulation.

Hence, we can write

$$\begin{aligned} \frac{\partial Y_{approx}}{\partial \mathbf{x}_i} &= \int \cdots \int_{\mathcal{P}} g(\mathbf{z}) \frac{\partial \Phi_{\mathbf{X}}(\mathbf{z})}{\partial x_i} d\mathbf{z} \\ &= \int \cdots \int_{\mathcal{P}} \left[\frac{g(\mathbf{z})}{\Phi_{\mathbf{X}}(\mathbf{z})} \frac{\partial \Phi_{\mathbf{X}}(\mathbf{z})}{\partial \mathbf{x}_i} \right] \Phi_{\mathbf{X}}(\mathbf{z}) d\mathbf{z}. \end{aligned} \tag{25}$$

Therefore, the corresponding yield estimator, based on a sample of N points, is taken as

$$\widehat{\frac{\partial Y}{\partial \mathbf{x}_i}} = \frac{1}{N} \sum_{k=0}^{N} \frac{g(\mathbf{z}_k)}{\Phi_{\mathbf{X}}(\mathbf{z}_k)} \frac{\partial \Phi_{\mathbf{X}}(\mathbf{z}_k)}{\partial \mathbf{x}_i}. \tag{26}$$

E. Effect of Monte Carlo Noise

If the yield of the circuit is small, then Monte Carlo noise is liable to affect the quality of the results of the convex programming algorithm. However, this may be offset by increasing the number of Monte Carlo points. Since the yield is evaluated by finding out the number of Monte Carlo points that lie within the approximating polytope, the additional computational expense in evaluating the yield and the yield gradient is relatively insignificant even though the number of Monte Carlo points is increased. This is in contrast with the case where one would have to perform a larger number of circuit simulations to obtain a more accurate estimation of the yield.

IV. EXPERIMENTAL RESULTS

In this section, the following examples are presented :

- An ellipsoidal feasible region, to show the correctness of our algorithms.
- A triangular feasible region, to show the sensitivity of the design center to probability distributions.
- A low-pass filter [13].
- A high-pass filter, used as a circuit example in [2]; yield comparisons of the design centers obtained using our methods and that in [2] are provided.
- A tunable active filter, used as a circuit example in [2]. As before, yield comparisons of the design centers are presented.
- A band-stop filter [14].
- A CMOS operational amplifier [15]

Many of these examples have been presented in earlier papers on design centering and related topics. In particular, we have provided results on *all* of the circuit examples in Abdel-Malek *et al.*'s paper on design centering [2].

The variances in each of the circuit examples are in terms of percentages of a nominal value that corresponds to the design center found by the L.H.E. algorithm.

A. A Numerical Example

A numerical example in two dimensions is presented here to illustrate our technique. The feasible region is represented by the ellipse

$$\mathcal{F} = \left\{ \mathbf{z} \mid (\mathbf{z} - \mathbf{t})^T \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} (\mathbf{z} - \mathbf{t}) \leq 1 \right\} \tag{27}$$

with the ellipse center, $\mathbf{t}$ at $[4, 4]^T$. The approximating polytope, $\mathcal{P}$, shown in Fig. 1, can be seen to be a good approximation of the ellipse.

The largest Hessian ellipsoid algorithm (L.H.E.) in Section III B for finding the exact design center converged in two iterations to the point $[3.992, 4.008]^T$.

The convex programming-based algorithm (C.P.) in Section III C was used, with the variances of the two random variables being set to 1.0. As pointed out in Section III A, the specific values of the variances are immaterial to the solution of this problem, since the random variables are Gaussian and the feasible region is an ellipsoid. The algorithm required three iterations to converge to the point $[4.007, 4.019]^T$.

In both cases, a very good approximation to the design center, $[4, 4]^T$ was obtained.

B. The Importance of Using Probability Information

The following example illustrates the deficiencies of ellipsoidal-based approaches, such as simplicial approximation [4], the ellipsoidal approach of Abdel-Malek *et al.* [2], and our largest Hessian ellipsoid approach. This example provides an exposition of the need for using information on the probability distributions of the design variables.

The feasible region, $\mathcal{F}$, illustrated in Fig. 2, is a triangle in $\mathbf{R}^2$, described by the planes

$$\begin{aligned} x_1 + x_2 &\leq 1 \\ x_1 - x_2 &\geq -1 \\ x_2 &\geq 0. \end{aligned} \tag{28}$$

Any approach that attempts to inscribe the largest circle or ellipsoid in this region would compute the centroid, i.e. the center of the incircle of the triangle, $[0, \sqrt{2} - 1]^T$ as the design center.

For a simple case in which the variances of the Gaussian random variables, x_1 and x_2, are the same, it is easily seen that the center of area, $\left[0, \frac{1}{3}\right]^T$, is the design center. Thus, the solution found by fitting the largest circle or ellipsoid is not necessarily the true design center.

More precisely, it can be seen from the definition (Equation (11)) that the design center is a function of the probability distributions. Table 1 shows how our convex programming approach finds the design center of the triangle in this example, for various sets of parameter variances for the Gaussian random variables $\mathbf{x}_1$ and $\mathbf{x}_2$.

We observe from the table above that the $\mathbf{x}_1$ coordinate of the design center is virtually at 0, and is insensitive to the variance. This is not unexpected, since the triangle is symmetric about the line $\mathbf{x}_1 = 0$; hence, the $\mathbf{x}_1$ coordinate of the design center would be 0, regardless of the variances.

TABLE 1 : YIELD MAXIMIZATION USING THE CONVEX PROGRAMMING APPROACH
RESULTS FOR VARIOUS SETS OF PARAMETER VARIANCES

$[\sigma_{\mathbf{x}_1}, \sigma_{\mathbf{x}_2}]$	$\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2}$	Computed Design Center $(\mathbf{x}_1, \mathbf{x}_2)$
$[0.1, 0.5]$	0.20	$(-0.01, +0.47)$
$[0.2, 0.5]$	0.40	$(-0.03, +0.45)$
$[0.3, 0.5]$	0.60	$(+0.02, +0.41)$
$[1.0, 1.0]$	1.00	$(+0.05, +0.35)$
$[1.0, 0.8]$	1.25	$(+0.03, +0.34)$
$[1.0, 0.5]$	2.00	$(-0.01, +0.34)$
$[1.5, 0.5]$	3.00	$(-0.01, +0.34)$
$[3.0, 0.5]$	6.00	$(-0.01, +0.32)$
$[5.0, 0.5]$	10.0	$(-0.06, +0.31)$
$[4.0, 0.2]$	20.0	$(-0.04, +0.29)$

However, changes in the variance of the random variables do affect the $\mathbf{x}_2$ coordinate of the design center. When $\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2}$ is low, the ellipsoidal approximation is a good one. In the limit as $\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2} \rightarrow 0$, the design center tends to the center of the largest ellipsoid that can be inscribed in the triangle. When $\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2} = 1$, the design center is the centroid of the triangle. As $\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2}$ increases, the design center starts moving downward along the $\mathbf{x}_2$ axis. When $\sigma_{\mathbf{x}_1}/\sigma_{\mathbf{x}_2} \rightarrow \infty$, the design center tends to the origin.

C. Design Centering for a Low Pass Filter

This is an example of a low-pass filter [13], whose circuit diagram and specifications are shown in Fig. 3. The frequencies of interest are $\{0.45, 0.50, 0.55, 0.60, 1.0, 2.5\}$ rad/sec, which correspond to six constraints.

The parameter $\bar{I}_{loss}$ is defined as follows:

$$I_{loss}(j\omega) \triangleq 20\, log_{10} \left| \frac{V_1}{V_2}(j\omega) \right| \quad \text{dB} \tag{29}$$

$$\bar{I}_{loss}(j\omega) \quad \triangleq \quad I_{loss}(j\omega) - I_{loss}(j\omega_0) \quad \text{dB}. \tag{30}$$

For this example, the reference frequency, ω_0, is 0 rad/sec. The design parameters are $[C, L_1, L_2]$. The initial feasible point was taken as $[1.05\text{F}, 1.6\text{H}, 1.6\text{H}]$.

A comparison of the results obtained using the largest Hessian ellipsoid (L.H.E.) approach of Section III B and the convex programming (C.P.) approach of Section III C is shown is Table 2. The L.H.E. method found the point $[0.96039\text{F}, 1.84915\text{H}, 1.83961\text{H}]$ to be the design center. The tabulated results show the yield about this design center for various sets of parameter tolerances, and the design center and yield for the C.P. approach. The yield figures are based on a Monte Carlo simulation of 500 points, with the mean value at the calculated design center.

In this example, a noticeably better solution to the design centering problem is given by the convex programming approach as compared to the ellipsoidal approximation-based method, for most of the cases listed above. This serves to bring out the fact that the center of an ellipsoid is not necessarily the design center for a circuit, and that a better solution can be obtained by incorporating information about the probability density functions that describe the design parameters.

TABLE 2 : YIELD MAXIMIZATION FOR A LOW PASS FILTER

Parameter Variations $[\sigma_C, \sigma_{L_1}, \sigma_{L_2}]$	C.P. Design Center in [F, H, H]	Yield (500 max)	
		C. P.	L. H. E.
$[5.0, 5.0, 5.0]$	$[0.8983, 2.1371, 1.8153]$	495	488
$[8.0, 8.0, 8.0]$	$[0.9359, 2.0464, 1.7779]$	457	452
$[10.0, 10.0, 10.0]$	$[0.9441, 2.0369, 1.7331]$	394	395
$[13.0, 13.0, 13.0]$	$[0.9485, 2.0427, 1.7079]$	324	319
$[8.0, 10.0, 15.0]$	$[0.9339, 2.1263, 1.6883]$	389	393
$[10.0, 13.0, 8.0]$	$[0.9506, 1.9630, 1.7872]$	395	385
$[5.0, 10.0, 8.0]$	$[0.9001, 2.1214, 1.7642]$	476	458

D. Design Centering for a High Pass Filter

This is an example of a high-pass filter [2], whose circuit diagram and specifications are shown in Fig. 4. The frequencies of interest are $\{170, 350, 440, 630, 680, 990, 1800\}$ Hz, which correspond to seven constraints. The definition of the parameter $\bar{I}_{loss}$ is analogous to that for the previous example.

For this example, the reference frequency, ω_0, is 990 Hz. The design parameters are $[C_1, C_3, C_4, C_5]$. The initial feasible point was taken as [11.1nF, 12.9nF, 34.3nF, 97.3nF], as in Abdel-Malek *et al.*'s ellipsoidal method (A.-M.E.) [2], where the solution was found to be [10.37nF, 13.28nF, 34.63nF, 87.84nF]. The solution found by the L.H.E. method is the point [9.614nF, 14.148nF, 33.642nF, 99.301nF]. The C.P. method was applied and, as before, the design center was found to change, depending on the variances associated with the p.d.f.'s.

TABLE 3 : YIELD MAXIMIZATION FOR A HIGH PASS FILTER (FOUR PARAMETERS)

Parameter Variations $[\sigma_{C_1}, \sigma_{C_3}, \sigma_{C_4}, \sigma_{C_5}]$	Yield (500 max)		
	L.H.E.	A.-M.E.	C.P.
[9.0, 9.0, 9.0, 9.0]	439	432	434
[10.0, 10.0, 10.0, 10.0]	412	407	415
[10.0, 15.0, 20.0, 5.0]	262	258	252
[15.0, 20.0, 5.0, 10.0]	432	403	422
[5.0, 8.0, 10.0, 12.0]	423	409	417
[5.0, 8.0, 10.0, 15.0]	411	400	404
[5.0, 10.0, 10.0, 15.0]	403	392	402
[5.0, 10.0, 12.0, 15.0]	365	365	364
[8.0, 12.0, 10.0, 12.0]	409	390	403
[8.0, 12.0, 10.0, 10.0]	411	399	416
[15.0, 12.0, 10.0, 10.0]	398	387	395
[8.0, 12.0, 12.0, 10.0]	373	360	369

TABLE 4 : YIELD MAXIMIZATION FOR A HIGH PASS FILTER (SEVEN PARAMETERS)

Parameter Variations $[\sigma_{C_1}, \sigma_{C_2}, \sigma_{C_3}, \sigma_{C_4}, \sigma_{C_5}, \sigma_{L_1}, \sigma_{L_2}]$	Yield (500 max)		
	L.H.E.	A.-M.E.	C.P.
[10, 10, 10, 10, 5, 5, 5]	243	244	251
[10, 10, 10, 10, 10, 10, 10]	159	149	155
[8, 8, 8, 8, 8, 8, 8]	237	220	231
[5, 5, 5, 5, 5, 5, 5]	420	363	410
[8, 8, 8, 8, 5, 5, 5]	307	303	306
[5, 10, 5, 10, 5, 10, 5]	243	205	230
[8, 12, 10, 12, 5, 8, 8]	175	162	164
[4, 4, 8, 8, 8, 4, 4]	361	359	363
[5, 5, 5, 5, 10, 10, 10]	257	228	246
[9, 10, 8, 10, 5, 4, 6]	256	253	260

The yield figures for the design centers found by the A.-M.E. method, the L.H.E. method and the C.P. method are displayed in Table 3. As before, the figures are based on a Monte Carlo simulation of 500 points, with the mean at the calculated center. It can be seen from this table that both the L.H.E. and the C.P. methods frequently performed better than the A.-M.E. method, and none of these three methods is consistently better than another. However, the overall performance of L.H.E. is the best of the three for this example.

The second experiment took $[C_1, C_2, C_3, C_4, C_5, L_1, L_2]$ as the design parameters. The initial feasible point was taken, as in [2], as [11.65 nF,10.47 nF,13.99 nF,39.93 nF,99.4 nF,3.988 H,2.685 H], where the solution was found to be [12.76 nF,10.37 nF,11.88 nF,40.26 nF,117.37 nF,3.609 H,2.504 H]. The solution obtained by the L.H.E. method was [12.68 nF,8.775 nF,12.68 nF,30.93 nF,93.17 nF,4.623 H,2.748 H]. A comparison of results for this experiment from the L.H.E., C.P. and A.M.-E. methods, for variance values of component variances, is shown in Table 4. Both the C.P. and L.H.E. methods provide better solutions than A.-M.E. for almost all cases here.

E. Design Centering for a Tunable Active Filter

The techniques described in the paper were used to design the tunable active filter shown in Fig. 5 [16]. The transfer function is given by

$$\mathbf{F} = \left| \frac{v_2}{v_g} \right| . \tag{31}$$

TABLE 5 : YIELD MAXIMIZATION FOR A TUNABLE ACTIVE FILTER

Parameter variances $[\sigma_{G_1}, \sigma_{G_4}, \sigma_{C_1}, \sigma_{C_2}]$	C.P. Design Center in $[\mu\mho, \mathrm{m}\mho, \mu\mathrm{F}, \mu\mathrm{F}]$	Yield (500 max)		
		C.P.	L.H.E.	A.-M.E.
[1.00, 1.00, 1.00, 1.00]	[83.1537,5.5237,0.7051,0.7652]	334	340	334
[1.50, 1.50, 1.50, 1.50]	[82.4903,5.6678,0.7277,0.7619]	226	233	229
[1.50, 1.50, 1.50, 0.50]	[82.9126,5.6594,0.7200,0.7677]	268	268	264
[1.50, 1.50, 2.00, 0.50]	[83.1716,5.6420,0.7165,0.7694]	229	236	228
[0.50, 0.80, 1.00, 0.80]	[83.9927,5.5682,0.7003,0.7745]	380	375	380
[0.50, 0.80, 0.70, 0.60]	[79.5042,5.4012,0.7219,0.7323]	431	432	432
[1.00, 0.40, 1.00, 0.60]	[79.5042,5.4012,0.7219,0.7323]	415	421	413
[1.00, 1.20, 1.00, 0.60]	[81.5437,5.4771,0.7134,0.7509]	357	360	362

The one pole role-off model used for the operational amplifier presumes a dc gain of 2×10^5, and a 3-dB bandwidth of 12π rad. The designable parameters are considered to be $[G_1, G_4, C_1, C_4]$. The

specifications for the filter are as follows:

$$\begin{array}{ccccccc} & & \mathbf{F} & \leq & 0.5 & \text{at} & 90\text{ Hz} \\ & & \mathbf{F} & \geq & 0.5 & \text{at} & 92\text{ Hz} \\ 1.0 & \leq & \mathbf{F} & \leq & 1.21 & \text{at} & 100\text{ Hz} \\ & & \mathbf{F} & \geq & 0.5 & \text{at} & 108\text{ Hz} \\ & & \mathbf{F} & \leq & 0.5 & \text{at} & 110\text{ Hz} \end{array} \tag{32}$$

An initial feasible point [80.2 $\mu\mho$, 5.42 m$\mho$, 0.72856 μF, 0.72856 μF] is provided, as in [2]. Table 5 shows a comparison of the results from the two approaches in this paper, namely the largest Hessian Ellipsoid (L.H.E.) approach and the convex programming (C.P.) approach in comparison with the result from the A.-M.E. approach, where the design center was calculated to be [83.13 $\mu\mho$, 5.423 m$\mho$, 0.716047 μF, 0.769181 μF] [2]. The L.H.E. method computed the point [84.29 $\mu\mho$, 5.636 m$\mho$, 0.71798 μF, 0.78037 μF] as the design center. The table compares the yield for a design centered about the two points above, and with the design center obtained by using the C.P. approach for each set of variances. The variances are given as percentages of the initial feasible point. The yield figures represent the number of feasible points in a Monte Carlo simulation using 500 sample points, with the mean fixed to be the computed design center.

All three methods provide approximately the same quality of result.

F. Design of a Band Stop Filter

This example of a band-stop filter is taken from [14]. A description of the circuit and the band diagram for the filter is provided in Fig. 6. The designable parameters are chosen to be $[C_1, C_2, C_3, L_1, L_2, L_3]$, with the initial values of $[22.14\text{pF}, 66.31\text{pF}, 22.14\text{pF}, 11.94\mu\text{H}, 3.986\mu\text{H}, 11.94\mu\text{H}]$.

TABLE 6 : YIELD MAXIMIZATION FOR A BAND STOP FILTER

Parameter variances $[\sigma_{C_1}, \sigma_{C_2}, \sigma_{C_3}, \sigma_{L_1}, \sigma_{L_2}, \sigma_{L_3}]$	C.P. Design Center in [pF, pF, pF, μH, μH, μH]	Yield (500 max)	
		C.P.	L.H.E.
[2.0, 2.0, 2.0, 2.0, 2.0, 2.0]	[22.4190, 68.3267, 22.1863, 11.3284, 4.0638, 11.5871]	240	244
[3.0, 3.0, 3.0, 3.0, 3.0, 3.0]	[22.1870, 68.0586, 21.8802, 11.5312, 4.0638, 11.6887]	130	126
[3.0, 2.0, 1.0, 3.0, 2.0, 1.0]	[22.2503, 68.1142, 22.1536, 11.5677, 4.0718, 11.3770]	257	240
[1.0, 2.0, 3.0, 1.0, 2.0, 3.0]	[22.6960, 68.2509, 21.7533, 11.1709, 4.0657, 11.8229]	261	238
[3.0, 1.0, 2.0, 3.0, 1.0, 2.0]	[22.4751, 67.6409, 20.9736, 11.4665, 4.0995, 12.0349]	322	306
[1.5, 1.0, 2.0, 3.0, 1.0, 1.5]	[22.2868, 68.3106, 21.5708, 11.4559, 4.0764, 11.8039]	330	337
[2.5, 2.0, 1.0, 2.0, 1.5, 1.5]	[22.3061, 67.8269, 22.4988, 11.4126, 4.0864, 11.4105]	270	273

The solution to the problem given by the L.H.E. method is [23.030pF, 66.542pF, 23.030pF, 11.085μH, 4.174μH, 11.085μH]. Table 6 shows the yield for the circuit, for different set of parameter variances, based on a Monte Carlo simulation using 500 sample points, with the mean fixed to be the design center computed by the L.H.E. and the C.P. methods, respectively. It is seen that each method performs better than the other an equal number of times in this table.

G. Design of a CMOS Operational Amplifier

The design centering techniques were applied to a CMOS Operational Amplifier circuit shown in Fig. 7 [15]. The transistor pairs M1-M2 and M3-M4 are matched. The designable parameters are the widths of transistors M1, M5, and M6. The constraints that define the feasible region for this problem are as follows:

- Gain $\geq$ 98 dB
- Area = sum of widths of M1, M5 and M6 $\leq$ 308
- Bandwidth $\geq$ 17 $\times 10^6$ rad/sec
- Power dissipation $\leq$ 0.65 mW

TABLE 7 : YIELD MAXIMIZATION FOR A CMOS OPERATIONAL AMPLIFIER

Parameter variances $[\sigma_{w_{M1}}, \sigma_{w_{M5}}, \sigma_{w_{M6}}]$	C.P. Design Center in [μm, μm, μm]	Yield (500 max)	
		C.P.	L.H.E.
[3.0, 3.0, 3.0]	[106.7,87.8,108.8]	452	446
[4.0, 4.0, 4.0]	[108.4,94.7,94.7]	392	377
[5.0, 5.0, 5.0]	[108.4,94.7,94.7]	334	312
[7.0, 7.0, 7.0]	[105.5,93.9,98.7]	252	219
[5.0, 8.0, 1.0]	[108.4,94.7,94.7]	302	281
[5.0, 7.0, 4.0]	[103.5 93.7 101.3]	300	288
[3.0, 4.0, 9.0]	[108.4,94.7,94.7]	288	289
[3.0, 6.0, 3.0]	[108.4,94.7,94.7]	370	375
[4.0, 1.0, 4.0]	[106.6,84.9,107.2]	425	411
[3.0, 2.0, 4.0]	[106.5,88.6,102.9]	440	437
[4.0, 2.0, 3.0]	[108.1,88.2,102.0]	445	421
[4.0, 4.0, 2.0]	[108.4,94.7,94.7]	433	413
[3.0, 5.0, 3.0]	[108.4,94.7,94.7]	411	401

We have assumed in this example, that the sizes of the three transistors vary independently of each other. In practise, however, this is not necessarily true as the size variations in an integrated circuit are correlated. However, the example is adequate to illustrate the design centering techniqes.

The nominal values of all transistors, corresponding to an initial feasible solution, are shown in the figure. We perform the L.H.E. and the C.P. design centering procedures in the transformed domain, to obtain the results shown in Table 7. As before, the yield estimates are based on a Monte Carlo simulation of 500 points with the design center corresponding to the mean value. It can be seen that for this example, the C.P. method is almost always better than the L.H.E. method.

V. CONCLUSION

In this paper, a new approach to design centering has been presented. The algorithm first approximates the feasible region by a polytope, using a procedure that is computationally less expensive than the existing method. Next, one of two approaches may be used to find the design center. The first approach inscribes the largest Hessian ellipsoid within the approximating polytope. This method is an improvement over simplicial approximation [4], where a hypersphere, or a crude ellipsoid are inscribed within the polytope, since the Hessian ellipsoid provides a good measure of the shape of the polytope. The second approach improves upon other geometrical design centering techniques, such as simplicial approximation, the ellipsoidal algorithm of Abdel-Malek *et al.* [2], and our first approach, by explicitly including information about the probability distributions of the random variables that represent the design parameters. The deficiencies of ellipsoid-based methods and the consequent need for a probability distribution-based approach have been illustrated by the example of the triangular feasible region in Section IV B. The low-pass filter in Section IV C is a practical example where an improvement in the solution has been obtained using this technique.

Circuit examples on a low-pass filter, a high-pass filter, a tunable active filter, a band-stop filter, and a CMOS op amp are presented. For each of these examples, the feasible region is nonconvex in reality. Many of these examples have been presented in earlier papers on design centering.

Comparisons with the results of the ellipsoidal method of Abdel-Malek *et al.* were made for all of the circuit examples published in [2], i.e., for a high-pass filter example and a tunable active filter example. For the high-pass filter, the L.H.E. (largest Hessian ellipsoid) and the C.P. (convex programming)

approaches provide solutions with better yields than the result of [2] in both the four-variable and seven-variable experiments. In the case of the tunable active filter, all three methods gave solutions of roughly the same quality, and our results were not noticeably worse than the results from A.-M.E.

In the case of the low-pass filter, the results of the C.P. method were better than those of the L.H.E. method for almost all cases, whereas for the band-stop filter, each method performed better than the other an equal number of times. For the example of the CMOS op amp, the C.P. method is almost always consistently better than the L.H.E. method.

The reason for the inconsistency in the results is that the quality of the results is affected by the convexity of the region. For convex feasible regions, the polytope approximation of the feasible region is more general than an ellipsoidal approximation, since it does not assume any symmetry in the structure of the feasible region. Hence, it would be expected that the C.P. method provides the best solution, followed by the L.H.E., A.-M.E. and simplicial approximation methods.

For nonconvex feasible regions, the quality of the geometrical methods deteriorates. The accuracy of these methods suffers since the polytope approximation to a nonconvex feasible region is poor. In particular, the basis of the convex programming approach lies in the correct approximation of the feasible region by a polytope, and in the absence of this correctness, the accuracy of the method suffers. Hence, in some cases in the examples above, this approach was found to be inferior to the other approaches. In practical examples, however, from experimental evidence, it is seen that both the L.H.E. and C.P. approaches are successful, regardless of whether the feasible region is convex or not.

ACKNOWLEDGEMENTS

The authors would like to extend their thanks to Abhijit Dharchoudhury for many helpful discussions. They would also like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[1] R. Spence and R. S. Soin, *Tolerance Design of Integrated Circuits.* Addison-Wesley, 1988.

[2] H. L. Abdel-Malek and A.-K. S. O. Hassan, "The ellipsoidal technique for design centering and region approximation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, pp. 1006–1014, Aug. 1991.

[3] P. M. Vaidya, "A new algorithm for minimizing convex functions over convex sets," *Proc. IEEE Foundations of Computer Science*, pp. 332–337, Oct. 1989.

[4] S. W. Director and G. D. Hachtel, "The simplicial approximation approach to design centering," *IEEE Transactions on Circuits and Systems*, vol. CAS-24, no. 7, pp. 363–372, 1977.

[5] P. Feldmann and S. W. Director, "Accurate and efficient evaluation of circuit yield and yield gradients," *Proceedings of the 1991 International Conference on Computer-Aided Design, Santa Clara, CA*, pp. 120–123, Nov. 1991.

[6] M. C. Bernardo, R. Buck, L. Liu, W. A. Nazaret, J. Sacks, and W. J. Welch, "Integrated circuit design optimization using a sequential strategy," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, Mar. 1992.

[7] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 1621–1634, Nov. 1993.

[8] T. Gao, P. M. Vaidya, and C. L. Liu, "A new performance driven placement algorithm," *Proceedings of the 1991 International Conference on Computer-Aided Design, Santa Clara, CA*, pp. 44–47, Nov. 1991.

[9] S. W. Director, W. Maly, and A. J. Strojwas, *VLSI Design for Manufacturing: Yield Enhancement.* Kluwer Academic Publishers, 1990.

[10] D. G. Luenberger, *Linear and Nonlinear Programming.* Addison-Wesley, 1984.

[11] L. O. Chua and P. M. Lin, *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques.* Prentice-Hall, 1975.

[12] A. Prekopa, "Logarithmic concave measures and related topics," in *Stochastic Programming* (M. Dempster, ed.), pp. 63–82, Academic Press, London, 1980.

[13] J. W. Bandler, P. C. Liu, and H. Tromp, "Non-linear programming approach to optimal design centering, tolerancing and tuning," *IEEE Transactions on Circuits and Systems*, vol. CAS-23, Mar. 1976.

[14] S. Niewiadomski, *Filter Handbook.* Heinemann Newnes, 1989.

[15] P. R. Gray and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits.* John Wiley and Sons, 1984.

[16] J. W. Bandler, H. L. Abdel-Malek, P. Dalsgaard, A. S. El-Razaz, and M. R. M. Rizk, "Optimization and design centering fo active and nonlinear circuits including component tolerances and model uncertainties," *Proceedings of the International Symposium on Large Scale Engineering Systems*, pp. 127–132, May 1978.