System Design by Constraint Adaptation and Differential Evolution

by Rainer Storn[1)]

TR-96-039

November 1996

Abstract

A simple optimization procedure for constraint based problems which works without an objective function is described. The absence of an objective function makes the problem formulation particularly simple. The new method lends itself to parallel computation and is well suited for tasks where a family of solutions is required, trade-off situations have to be dealt with or the design center has to be found.

[1)]International Computer Science Institute, 1947 Center Street, Berkeley, CA 94704-1198, Suite 600, Fax: 510-643-7684. E-mail: storn@icsi.berkeley.edu. On leave from Siemens AG, ZFE T SN 2, Otto-Hahn-Ring 6, D-81739 Muenchen, Germany. Fax: 01149-636-44577, E-mail:rainer.storn@zfe.siemens.de.

1. Introduction

The design of a technical system is usually associated with the process of properly choosing some system parameters such that the technical system meets its specifications. The parameter choosing process can also be regarded as an optimization where the goal is to minimize some error function. The error function, or cost function, has to measure somehow the extent to which the specifications are violated. Take for example the task of choosing the real and continuous parameters p_1 and p_2 of a function

$$f_{p_1,p_2}(x) = \frac{1}{1 + p_1 x + (p_2 x)^2}, \tag{1}$$

so that $f_{p_1,p_2}(x)$ fits into the tolerance scheme depicted in Fig. 1a).

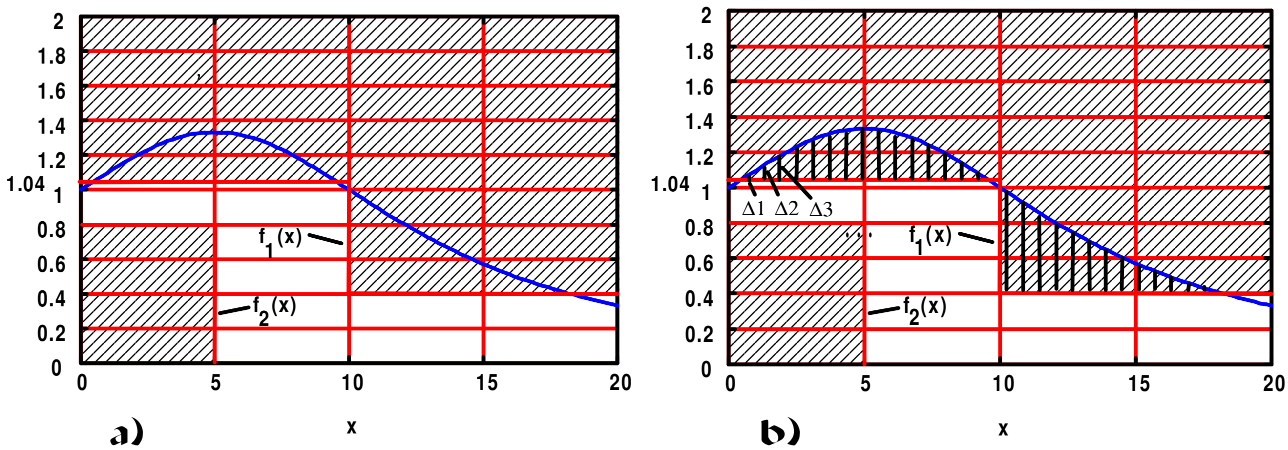


Fig. 1: a) For p_1=-0.1 and p_2=0.1 $f_{p_1,p_2}(x)$ does not meet the requirement to avoid the grey shaded areas.
b) The extent to which $f_{p_1,p_2}(x)$ violates the requirements can be measured by the errors Δi, i=1,2, ... , N, taken, for example, at equidistant abscissa values.

The goal of function $f_{p_1,p_2}(x)$ is to stay apart from the shaded areas and run between the two non-differentiable constraint functions $f_1(x)$ and $f_2(x)$. The above kind of design problem shows up e.g. in electronic filter theory, where a transfer function of a lowpass filter ought to let the low frequencies of an input signal pass while the high frequencies shall be supressed sufficiently. Another application where functions have to fit a tolerance scheme is optimal control theory where the trajectory of a vehicle has to stay within certain limits, so that the vehicle doesn't collide with surrounding obstacles. The standard way to solve the general problem of fitting an arbitrary function into a tolerance scheme is to contrive an objective function the minimization of which leads to the appropriate choice of function parameters. An example of such an objective function ist depicted in Fig. 1b).

The objective function could be

$$e_{p_1,p_2}(x) = \sum_{i=1}^{N} \Delta i^2 \tag{2}$$

with
$$\Delta i = \begin{cases} f_{p_1,p_2}(x_i) - f_1(x_i) & \text{if} \quad f_{p_1,p_2}(x_i) > f_1(x_i) \\ f_{p_1,p_2}(x_i) - f_2(x_i) & \text{if} \quad f_{p_1,p_2}(x_i) < f_2(x_i). \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

and the surface plot of which is shown in fig. 2. The minimization of $e_{p_1,p_2}(x)$ will yield the parameters p_1, p_2 for which $f_{p_1,p_2}(x)$ stays within the prescribed limits, if the problem has a solution. In such a case $e_{p_1,p_2}(x)$ will be zero. Otherwise the minimization of $e_{p_1,p_2}(x)$ renders a solution which is best according to the squared error criterion. Of course, other objective functions are also possible and, depending on the application, might even be more appropriate.

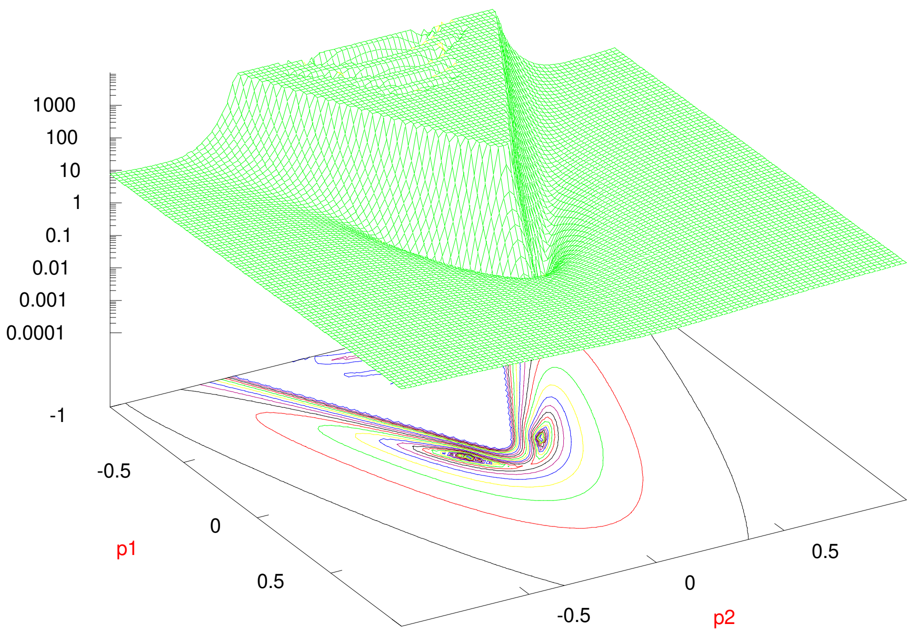


Fig. 2: Objective function for the problem in fig. 1 following eqs. (2) and (3).

Finding a pertinent objective function is crucial to the design process and gets increasingly complicated if

a) multiple objectives are pursued. In such a case the different objectives have to be weighted. Finding the proper weights can be a problem of its own.

b) the goal is to find the so called design center. This problem appears, for example, in electronic circuit design where single electronic components like e.g. resistors, capacitors or MOS transistor gate widths represent the parameters and are subject to a certain jitter due to imprecisions in the manufacturing process of these components. The goal in design centering is to choose the parameters such that maximum jitter is allowed and hence production yield is maximized.

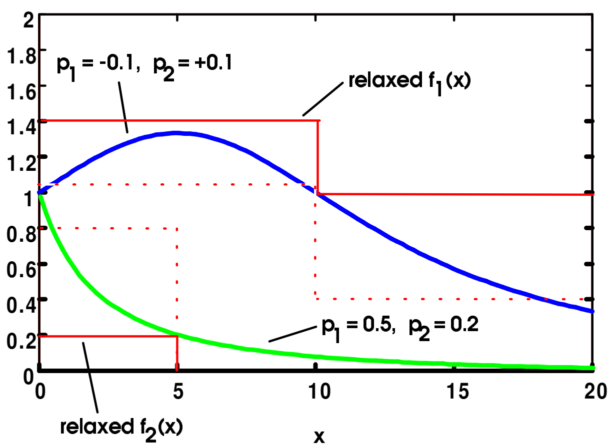
c) the design problem is heavily constraint based, especially if the constraints are nonlinear. Objective function approaches have to incorporate constraints via weighted penalty terms [1] which pose additional difficulties for the designers.

A design approach via an objective function might also be inappropriate if a family of solutions is required. In trade-off situations the system designers often want to have several options to choose from. An objective function based approach, however, generally yields only one solution.

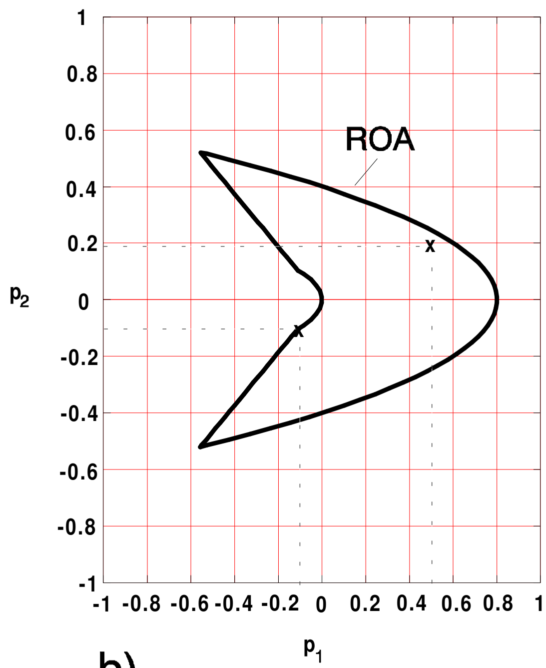
2. Constraint Adaptation

An alternative approach to the design via objective functions is constraint adaptation [2], [3] which is illustrated in Fig. 3. Instead of devising an objective function, the specifications of the design problem, in our example $f_1(x)$ and $f_2(x)$, are relaxed at the start of the design process. This is done in such a way that function $f_{p_1,p_2}(x)$ with its current choice of parameters fits into the relaxed tolerance scheme.

As indicated in Fig. 2a) there might also be more than one parameter set, corresponding to a family of functions $f_{p_1,p_2}(x)$, which satisfy the relaxed tolerance scheme. The specifications of the property domain, represented by the tolerance scheme, correspond to a so called region of acceptability (ROA) in the parameter domain . If a parameter set lies within the ROA the corresponding function meets the specifications. If it lies outside,



a)



b)

Fig. 3: Relaxed tolerance scheme in the property domain a) and corresponding region of acceptability (ROA) in the parameter domain b).

the specifications are violated. The goal is to find parameter sets which do not lie on the rim of the ROA but truly inside such that the ROA can be shrinked. Shrinking of the ROA goes along with a tightening of the specifications. The design process is performed by continuously shrinking the ROA until the relaxed specifications have tightened to the original ones. Specifications can be modeled as must-constraints and may-constraints. Must-constraints have to satisfy certain values, otherwise the design task is not successful. May-constraints, however, are improved on a best effort basis only. Design techniques which make use of constraint adaptation have been around for quite some time, especially in the circuit design community, [3] ... [6]. The basic idea there is to start with some nominal parameter set which is described as a parameter vector

$$\underline{p} = (p_1, p_2, \ldots, p_D)^T \tag{4}$$

for which the specifications are relaxed just enough to have the endpoint of $\underline{p}$ lying on the rim of the ROA. This situation is illustrated in Fig. 4. As the ROA is generally not known in advance, the region of acceptability has to be estimated. This estimation is done by generating a cloud of points around the nominal vector $\underline{p}$, usually via a multi-variate normal distribution as indicated in Fig. 4. Each point requires the system in question to be analyzed and checked whether the parameter vector lies inside or outside the current ROA. Only those parameter vectors which lie inside the ROA, the "hits", are further taken into consideration. The "no-hits" are discarded. The mean value of the "hits" is taken to serve as the new nominal vector. Eventually the ROA and correspondingly the relaxed specifications are tightened such that the new nominal vector lies on the rim of the new ROA etc..

There have been several modifications and improvements to this procedure [2], [3], yet there are still several problems inherent in the strategy described so far:

a) The mean value is an inappropriate choice for the new nominal vector if it lies outside the ROA. This can easily happen if the ROA is concave or consists of several disjunct islands.

b) If the ROA splits into disjunct islands the design method is in danger of getting trapped in one of the islands. It can very well happen that some of these islands vanish during a successful design process with only those islands remaining that provide a solution to the original problem. If, however, a vector gets trapped in an island that is bound to vanish, the design process stagnates.

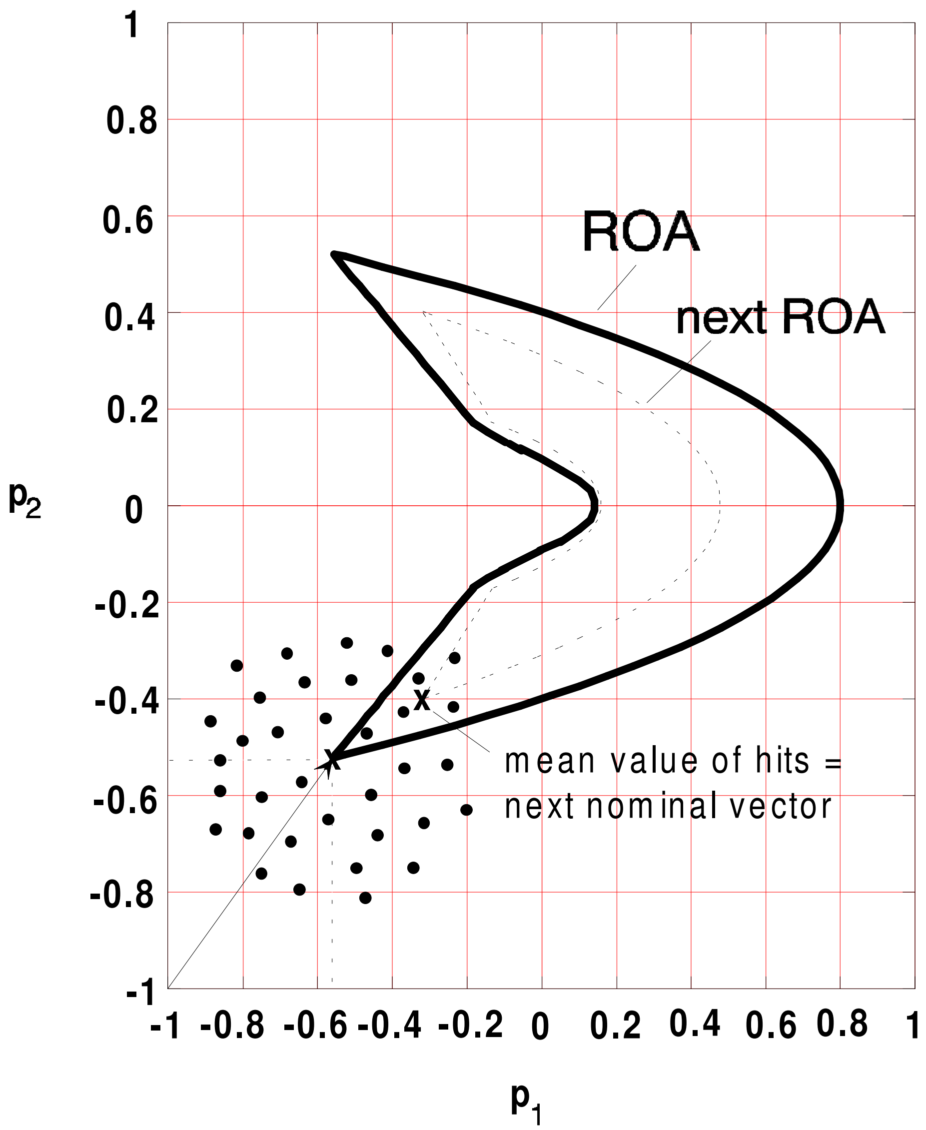


Fig. 4: The basic strategy for the conventional constraint adaptation procedure is to generate a cloud of points around a nominal vector and compute the new nominal vector as the mean value of the points inside the current ROA.

c) It is difficult to choose the standard deviations for the multivariate gaussian distribution. Most strategies have to fight against the standard deviations becoming too small.

d) The number of quadrants which the cloud of points potentially has to cover is 2^D, with D being the number of parameters. Hence the number of quadrants increases exponentially with the dimension D of the parameter space, requiring a large number of points in the cloud to increase the chance that "hits" will be generated.

3. Constraint Adaptation by Differential Evolution (CADE)

A new variant of constraint adaptation which is based on Differential Evolution (DE) [1] , [7], [8], [9], [10] provides a solution to the problems a) through c) above. Problem d), however, is inherent to all statistical design methods and limits the problem size which can be tackled.

DE is a powerful and robust statistical method for objective function minimization and appeals, among other things, by its absence of a predefined probability density function (pdf). This way poblem c) is nonexistent. DE is also a completely parallel method, which doesn't make use of a single nominal vector but uses a population of equally important vectors. CADE combines the ideas of constraint adaptation and DE into a versatile design method and is described below.

CADE uses NP parameter vectors

$$\underline{p}_{i,G};\ i = 1, 2, \ldots, NP \tag{5}$$

in a generation G, with NP being constant over the entire design process. At the start of the procedure, i.e. generation G=1, the population of vectors is usually chosen randomly. As a rule, we will assume a uniform probability distribution for all random decisions unless otherwise stated. The specifications of the system are relaxed just as much that all NP vectors lie inside the ROA. Note that any realizability constraints, like e.g. geometrical constraints or maximum component values in integrated circuit design, can easily be incorporated into this formulation. For the following generation G+1, new vectors are generated according to

$$\underline{v}_{i,G+1} = \underline{p}_{i,G} + F \cdot (\underline{p}_{r_1,G} - \underline{p}_{r_2,G}) \tag{6}$$

for $i = 1, 2, \ldots, NP$

with $r_1, r_2 \in [1, NP]$, integer and mutually different

as well as F real and > 0.

The integers r_1 and r_2 are chosen randomly and should be different from the running index i. F is a real constant factor which controls the amplification of the differential variation $(\underline{p}_{r_1,G} - \underline{p}_{r_2,G})$ and usually takes on values between 0.1 and 1. Fig. 5 shows an example of the vectors which play a part in the vector generation process.

In order to increase the diversity of the new parameter vectors, crossover is introduced an example of which is depicted in fig. 6. There are several variants of crossover [7], [8], [9], [10], one of them is to form the vector

$$\underline{u}_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \ldots, u_{Di,G+1})^T \tag{7}$$

with

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & \text{for} \quad j = \langle n \rangle_D, \langle n+1 \rangle_D, \ldots, \langle n+L-1 \rangle_D + 1 \\ p_{ji,G} & \text{otherwise} \end{cases} \tag{8}$$

where the acute brackets $\langle\ \rangle_D$ denote the modulo function with modulus D. The starting index n in (8) is a randomly chosen integer from the interval [1,D]. The integer L is also drawn from the interval [1,D] with the probability $Pr(L>=\nu) = (CR)^{\nu-1}$, $\nu>0$. CR, the so called crossover constant, is taken from the interval [0,1] and constitutes a control variable in the design process. The random decisions for both n and L are made anew for each newly generated vector $\underline{u}_{i,G+1}$.

If $\underline{u}_{i,G+1}$ lies within the ROA the new population member $\underline{p}_{i,G+1}$ becomes $\underline{u}_{i,G+1}$. If $\underline{u}_{i,G+1}$ lies outside the ROA, the vector generation process (6) and (7) is repeated up to NT times. If $\underline{u}_{i,G+1}$ still lies outside the ROA, the old vector $\underline{p}_{i,G}$ is checked for how many generations it has already been reused. If the vector $\underline{p}_{i,G}$ has an age of less than NG generations, $\underline{p}_{i,G+1}$ will be set to $\underline{p}_{i,G}$. If $\underline{p}_{i,G}$ has survived already for NG generations this vector must die because of age and $\underline{p}_{i,G+1}$ will be set to some arbitrary vector $\underline{p}_{r3,G}$ from the current generation, with $r_3 \in [1,NP]$ and integer.

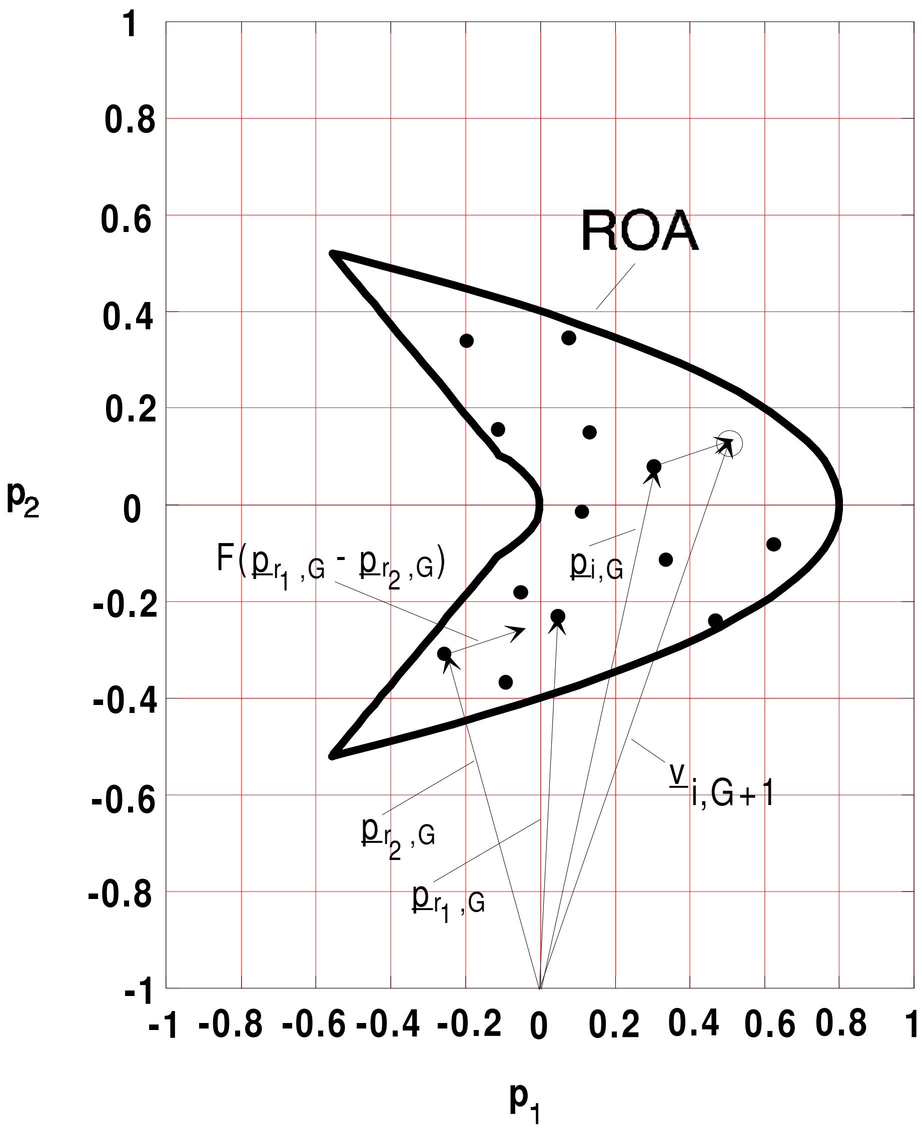


Fig. 5: Graphical illustration of the vector generation process (6).

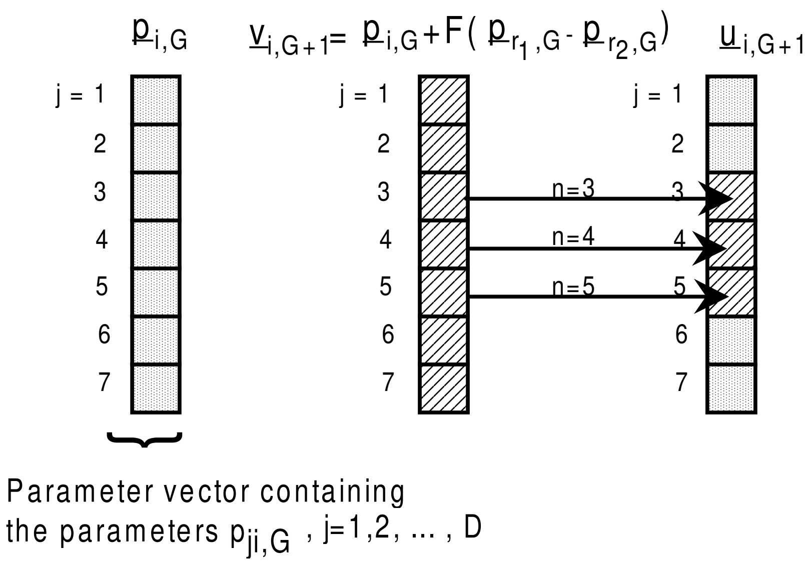


Fig. 6: Illustration of the crossover process for D=7, n=3 and L=3.

Once the new collection of vectors $\underline{p}_{i,G+1}$, i=1,2, ... ,NP, has been found, the ROA and its corresponding specifications for the system to be designed are tightened such that all vectors lie inside the new ROA and at least one of them lies on the rim of the new ROA. If the vector on the rim can reproduce in the following generation, it is very unlikely that its offspring lies on the rim of the current ROA. The same is true for vectors which already lie inside the ROA. Hence, the probability that the ROA shrinks from one generation to the other is high even though the shrinkage rate might be low.
The design process described is repeated until the design goals are met. If the goal is to find just one parameter set which meets the original specifications, the design procedure can be stopped as soon as one member of the vector population meets the requirements. This corresponds to the situation where the particular vector in question lies on the rim of the ROA which defines the original specifications. In case of trade-off situations and design centering tasks, the design procedure stops when all population members fulfill the specifications. If the feasibility of the design is not guaranteed or if there are restrictions on the total computing time for the design, the procedure might have to be stopped after a certain number of generations.
CADE lends itself perfectly to parallel computation which becomes increasingly important the more expensive a single sytem evaluation gets. In circuit design, a system evaluation is often done via a SPICE simulation [3], which can require a significant amount of computer time.

Several properties of CADE are worth elaborating upon. Except for a uniform random number generator which supplies pseudo random numbers from the interval [0,1] no specially shaped pdf is required and hence the difficulty of adapting the standard deviations of some multivariate pdf for each generation doesn't exist. Instead only two control variables, F and CR, have to chosen once for the entire design process and are taken from well defined intervals, i.e. [0.1,1] and [0,1] respectively. As an aside it shall be mentioned that the interval [0.1,1] for F can be extended, however, no additional benefits could be observed in the experiments that were performed.

Another important feature of CADE concerns the aging of the vectors. Empirical evidence has revealed that the introduction of aging is vital to the design process as it seems to happen fairly often that the ROA splits up into several disjunct islands. Fig. 7 shows an example of such a split-up for the design problem stated in Fig. 1.

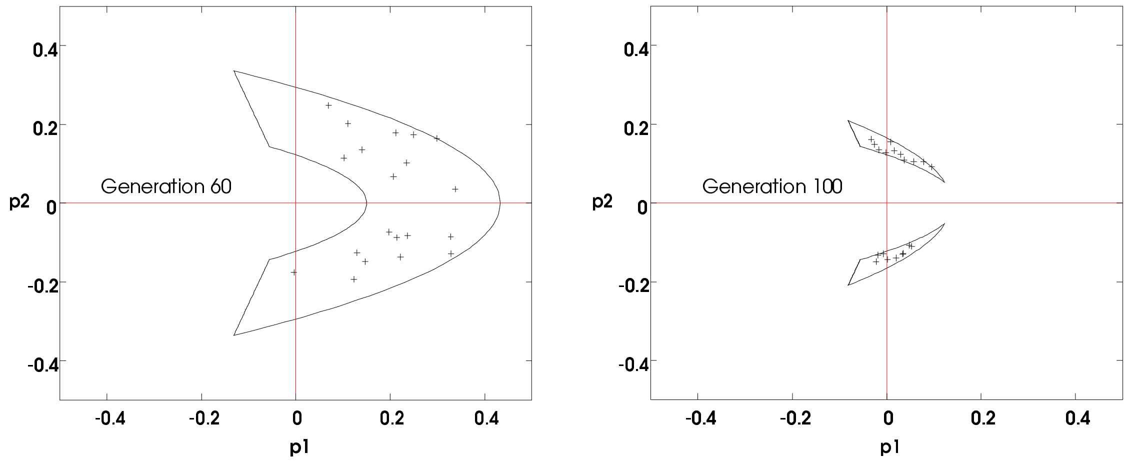


Fig. 7: Example for the split-up of an ROA as it occurs for the design problem stated in Fig. 1. The settings in CADE were NP=20, F=0.9, CR=0.3, NT=100 and NG=1.

If aging were not introduced, one vector could survive during the entire design process in an island that should vanish for the design process to be successful and henceforth induce stagnation. For the simple example in fig. 1 it has been found empirically that the design converged fastest if a vector is not allowed to survive more than NG=1 generation.

A further interesting property of CADE is the fact that the NP parameter vectors tend to diverge and most of them will occupy the most voluminous part of the ROA. We will refer to this property as "divergence and pooling". The following line of thought shall illustrate this property. Consider a cloud of points inside a ROA as indicated in Fig. 8. If the vectors have to reproduce according to the CADE scheme, it is very unlikely that the new population of vectors stays within the hull of the cloud made up by the current population. If, however, the cloud of points diverges, the mean distance between the points increases yielding even larger deviations for the next CADE step. The same argument holds for every generation. The only factor which restricts the divergence of the cloud is the ROA itself, as vectors which point outside the ROA don't constitute valid members of the population. The chance to reproduce is highest in the most voluminous portion of the ROA, so the vector population members tend to inhabit this region. Vectors which do not reside in this area are running a higher risk of dieing because of age. Fig. 9 shows the "divergence and pooling" property of CADE for a fixed ROA.

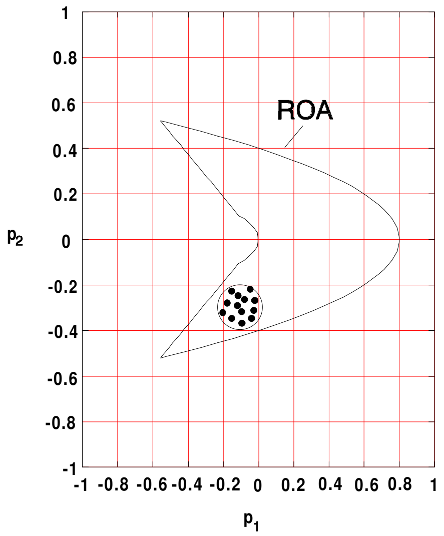


Fig. 8: If a vector population reproduces according to CADE, the offsprings can't be kept from passing the hull of the current cloud of points. This leads to the divergence property.

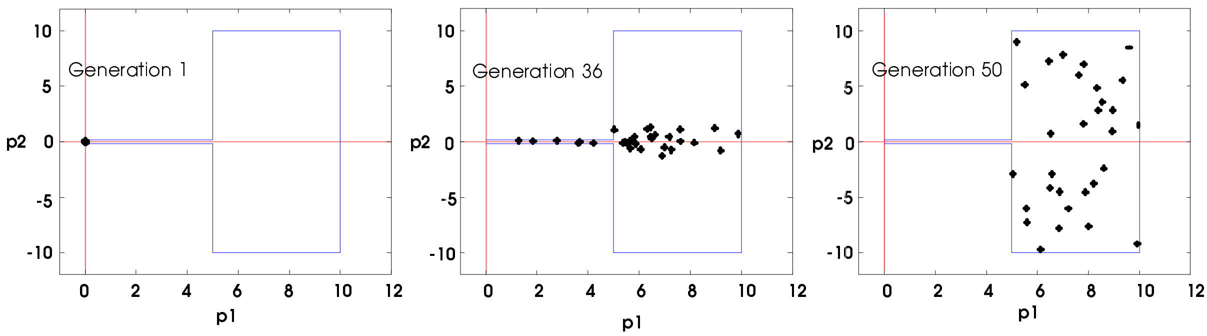


Fig. 9: Illustration of divergence and pooling of the vector population. The population started with NP=30 population members close to the origin. The control variables were F=0.9, CR=0.5, NT=10 and NG=1.

The divergence property of the vector population is very important for the CADE design process, as it helps to estimate the ROA over a wide region, not only in a local area. At the end of the design process the pooling effect is ideally suited for estimating the design center. The design center can be estimated by computing the mean value of all the population members.

The divergence property, however, also has its drawbacks as it impedes convergence. The number NP of population vectors influences convergence. A small number NP leads to a low number of system evaluations and also reduces the chance of offsprings lieing close to the rim of the ROA. Low numbers NP, however, yield a worse estimation of the ROA and decrease the chance of finding the desired solution.

4. Design Centering of an SC-Filter

Switched Capacitor (SC) Filters are the modern replacement for Resistor Capacitor (RC) Filters as they are better suited for integration on a silicon chip. This is due to the fact that the transfer function of

SC-Filters is mainly dependent on capacitor ratios which are less insensitive to geometrical errors in the integration process than are the RC time constants of RC-filters. The basic design procedures of SC-Filters and RC-Filters are essentially the same, the integration process, however, brings up SC-specific problems like parasitic capacitances which alter the transfer function. CADE is well suited to redesign an SC-Filter suffering from parasitic capacitances as will be demonstrated in the following. An example of a PCM-Lowpass is shown in fig. 10.

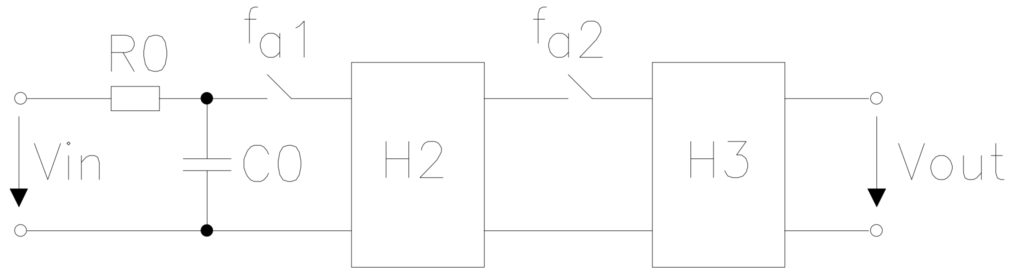


Fig. 10: Block diagram of the PCM-Lowpass filter.

The transfer function of the SC-PCM lowpass in Fig. 10 can be written as [11]

$$H(f) = \frac{Vout}{Vin} = H1(f) \cdot H2(w) \cdot H3(w) \tag{9}$$

with
$$w = j \cdot \tan\frac{\pi f}{f_a}, \tag{10}$$

and
$$j = \sqrt{-1}. \tag{11}$$

Frequency f_a is the sampling frequency in use. For H2(w) we choose $f_a = f_{a1}$=128kHz and for H3(w) $f_a = f_{a2}$=32kHz. For the analog RC-lowpass prefilter we get

$$H1(f) = \frac{1}{1 + j \cdot 2\pi f \cdot R0 \cdot C0} = \frac{1}{1 + j \cdot f \cdot v1}. \tag{12}$$

In [11] it has been derived that the other partial transfer functions may be written as

$$H2(w) = \frac{w^2 \cdot (v12 \cdot v32) + w \cdot (v12 + v32 - v132) + 1}{w^2 \cdot (v12 \cdot v32 + v12 \cdot v532) + w \cdot (v12 + v32 + v532 - v132) + 1} \tag{13}$$

and

$$H3(w) = \frac{w^2 \cdot (v13 \cdot v33) + w \cdot (v13 + v33 - v133) + 1}{w^2 \cdot (v13 \cdot v33 + v13 \cdot v533) + w \cdot (v13 + v33 + v533 - v133) + 1} \quad . \tag{14}$$

If we choose the parameter values in (12), (13) and (14) according to

v1 = 0.0005184;

v12 = 14.336;

v32 = 1.9957;

v132 = 16.332;

v532 = 11.813;

v13 = 6.9821;

v33 = 0.55987;

v133 = 7.5420;

v533 = 0.5223;

the magnitude of H(f) is represented by the graph in fig. 11.

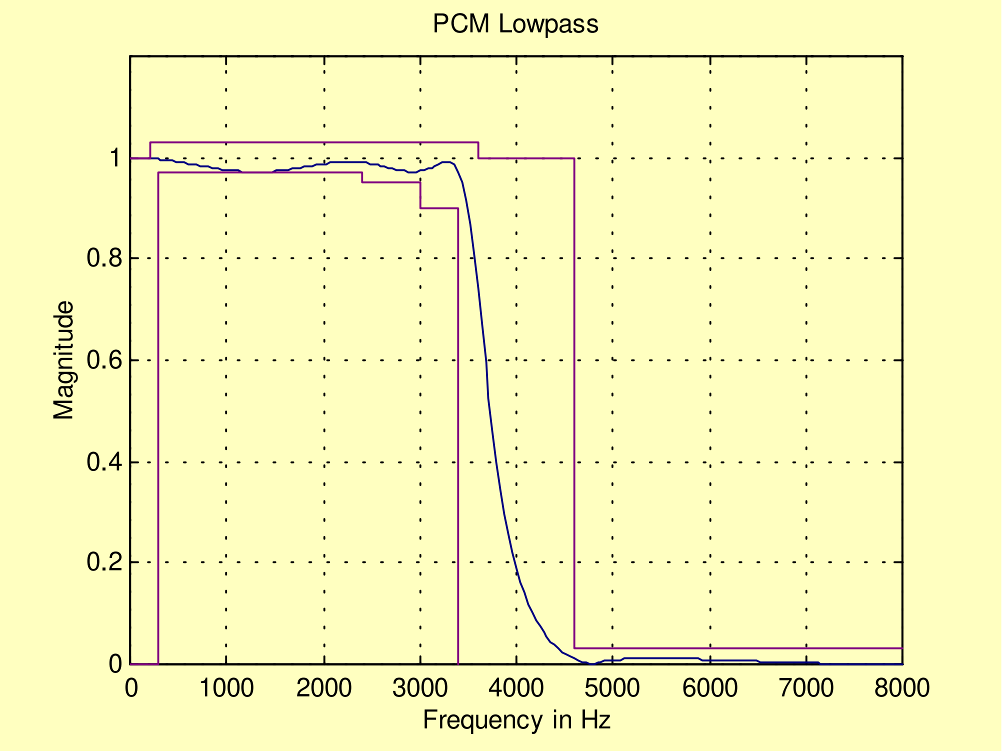


Fig. 11: |H(f)| for the SC-PCM lowpass without parasitic capacitances.

The design procedure fits the function into the tolerance scheme with upper constraints {1.0, 1.0292, 1.0, 0.031623} at frequencies {0,200,3600,4600} in Hz and the lower constraints {0.0, 0.97162, 0.94951, 0.90157, 0.0} at frequencies {0,300,2400,3000,3400} in Hz. If parasitic capacitances are taken into account, it can be shown [11] that, after some simplifying assumptions, the transfer functions H2(w) and H3(w) change into

$$H2(w) = \frac{w^2 \cdot \left[\left(v32 - \frac{\gamma}{2}\right)\left(v12(1+\varepsilon) + \frac{\gamma}{2}\right) + v132\frac{\gamma}{2}\right] + w \cdot \left[\left(1+\frac{\gamma}{2}\right)(v32 - v132) + v12(1+\varepsilon)\left(1+\frac{\gamma}{2}\right)\right] + \left(1+\frac{\gamma}{2}\right)^2}{w^2 \cdot \left[\left(v32 + v532 - \frac{\gamma}{2}\right)\left(v12(1+\varepsilon) + \frac{\gamma}{2}\right) + v132\frac{\gamma}{2}\right] + w \cdot \left[\left(1+\frac{\gamma}{2}\right)(v32 + v532 - v132) + v12(1+\varepsilon)\left(1+\frac{\gamma}{2}\right)\right] + \left(1+\frac{\gamma}{2}\right)^2} \quad (15)$$

and

$$H3(w) = \frac{w^2 \cdot \left[\left(v33 - \frac{\gamma}{2}\right)\left(v13(1+\varepsilon) + \frac{\gamma}{2}\right) + v133\frac{\gamma}{2}\right] + w \cdot \left[\left(1+\frac{\gamma}{2}\right)(v33 - v133) + v13(1+\varepsilon)\left(1+\frac{\gamma}{2}\right)\right] + \left(1+\frac{\gamma}{2}\right)^2}{w^2 \cdot \left[\left(v33 + v533 - \frac{\gamma}{2}\right)\left(v13(1+\varepsilon) + \frac{\gamma}{2}\right) + v133\frac{\gamma}{2}\right] + w \cdot \left[\left(1+\frac{\gamma}{2}\right)(v33 + v533 - v133) + v13(1+\varepsilon)\left(1+\frac{\gamma}{2}\right)\right] + \left(1+\frac{\gamma}{2}\right)^2}. \quad (16)$$

The constants $\gamma \in [2.55\%, 10.5\%]$ and $\varepsilon \in [0.1\%, 1\%]$ represent the parasitic effects. If we assume that the integration process under scrutiny is characterized by γ=5% and ε=0.5% and the parameter values for H1, H2, and H3 are not changed the parasitic influence changes |H(f)| according to fig. 12 which clearly shows that the tolerance scheme is now violated.

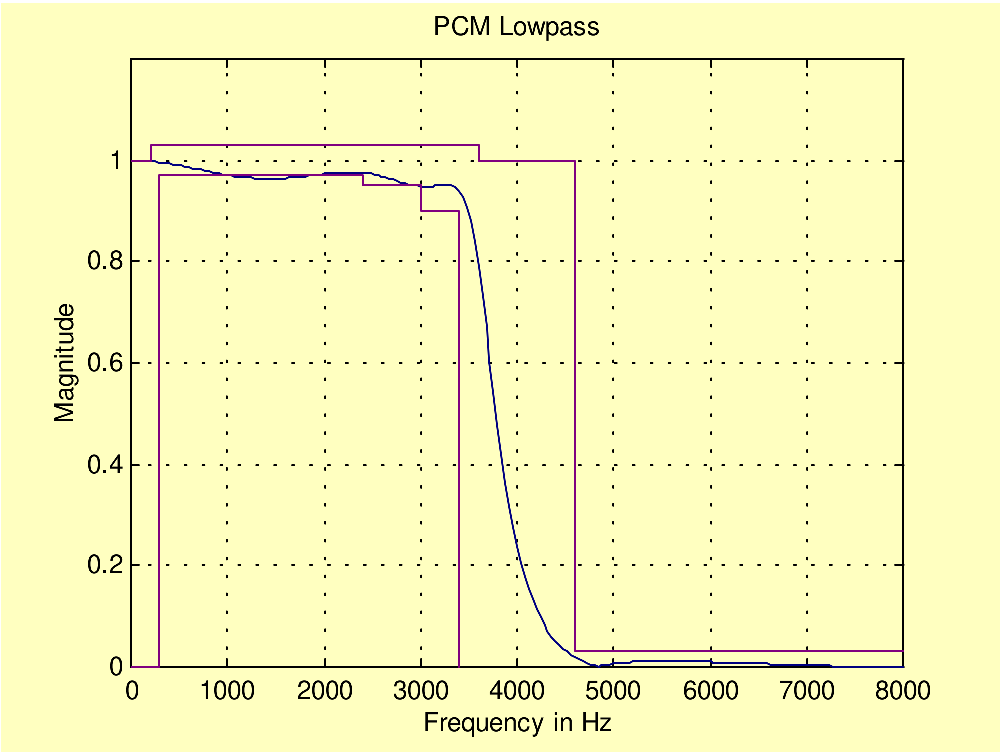


Fig. 12: |H(f)| after considering parasitic capacitances but without redesign.

In order to redesign the parameters such that the transfer function satisfies the above tolerance scheme, any nonlinear design procedure may be used. To center the design, however, so that the parameters may jitter maximally around their nominal value an approach like CADE which investigates the ROA is required.

In this example the steering variables NP=30, F=0.9, CR=1, NT=10 and NG=2 were chosen. The initial population was set up by randomly perturbing the original parameter values with a maximum deviation of 1%. After 4663 evaluations of |H(f)| the following parameter values emerged:

v1 = 0.000418;

v12 = 12.518721;

v32 = 2.780037;

v132 = 14.825224;

v532 = 12.326931;

v13 = 7.785936;

v33 = 0.537556;

v133 = 8.239942;

v533 = 0.475803;

The resulting magnitude plot in fig. 13 shows clearly that the design fits well into the tolerance scheme.

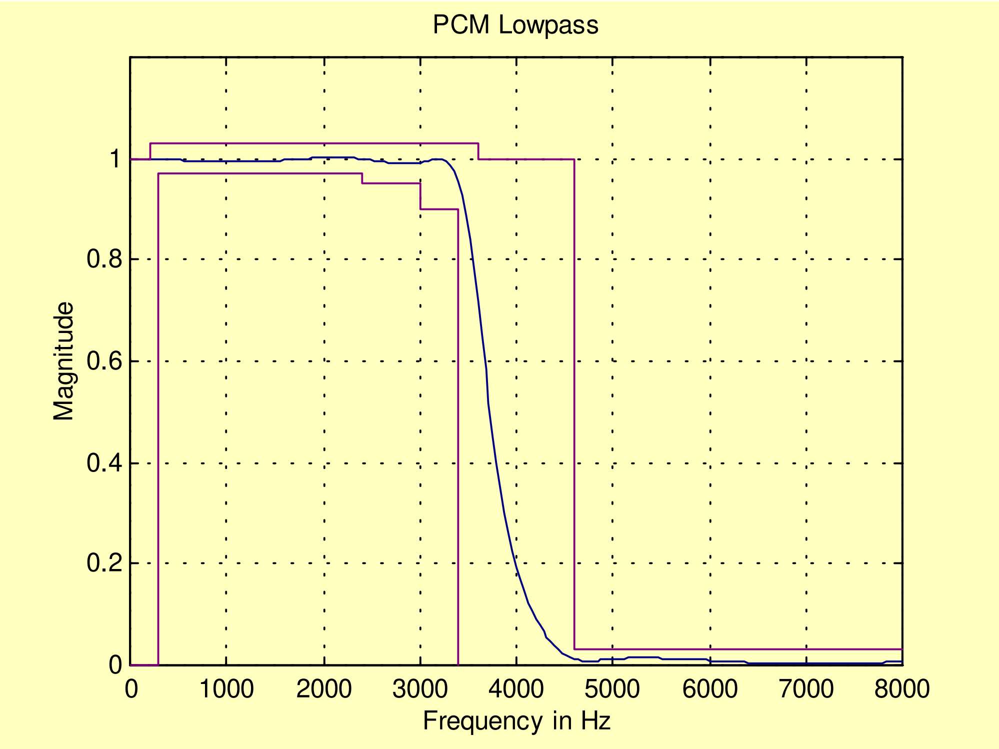


Fig. 13: |H(f)| after considering parasitic capacitances with redesigned parameters.

Although the SC-Filter example demonstrates the usability of CADE, several crucial points have to be addressed: it has been found that the parameter values above still change up to 1% if CADE keeps on running which corresponds to a non-settled mean value. This comes as no surprise as NP=30 population members represent very few points to sample an ROA of 9 dimensions. Consequently many more population members have to be used to get a mean value which eventually stabilizes. More population members, however, lead to an increased running time because more function evaluations have to be performed. In addition the shrinkage of the relaxed ROA to the original size is also slowed down because the shrinkage takes the weakest population member into account. With more population members the chance is increased that some members lie close to the rim of the ROA and hence reduce the shrinkage rate. Last but not least it has to be mentioned that design centering by statistical evaluation of the mean vector thrives upon the assumption that the ROA consists of only one island which, at best, is convex. This assumption might be or might not be true for a specific application. More research has to go into the design centering process via CADE to learn more about CADE's mechanisms and improve its performance.

5. References

[1] Storn, R. and Price, K., Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, ICSI.

[2] Storn, R., Constrained Optimization, Dr. Dobb's Journal, May 1995, pp. 119 - 123.

[3] Lueder, E., Optimization of Circuits With a Large Number of Parameters, Archiv fuer Elektr. und Uebertr., Vol. 44, Issue 2, 1990, pp. 131 - 138.

[4] Brayton, R., Hachtel, G. and Sangiovanni-Vincentelli, A., A Survey of Optimization Techniques for Integrated Circuit Design, Proc. of the IEEE 69, 1981, pp. 1334 - 1362.

[5] Eckstein, T. and Lueder, E., Design Centering by Improved Monte Carlo Analysis of the Region Of Acceptability, Proc. ISCAS, San Jose, 1986, pp. 951 - 954.

[6] Moebus, D. and Lueder, E., Optimization of Circuits Using a Combination of Deterministic and Statistical Search, Proc. ISCAS, 1987.

[7] Storn, Differential Evolution Design of an IIR-Filter with Requirements of Magnitude and Group Delay, IEEE Int. Conf. Evol. Comp., ICEC'96, pp. 268 - 273.

[8] Storn, R. and Price, K., Minimizing the Real Functions of the ICEC'96 Contest by Differential Evolution, IEEE Int. Conf. Evol. Comp., ICEC'96, pp. 842 - 844.

[9] Storn, R., On the Usage of Differential Evolution for Function Optimization, NAFIPS 1996, Berkeley, pp. 519 - 523.

[10] Price, K., Differential Evolution: A Fast and Simple Numerical Optimizer, NAFIPS 1996, Berkeley, pp. 524 - 527.

[11] Storn, R., Entwurfszentrierung und Aufbau einer Schaltung mit geschalteten Kondensatoren, Semesterarbeit am Institut f. Netzwerk- und Systemtheorie der Univ. Stuttgart, 1982.