# Metaheuristic Techniques for the Analog Circuits Performances Optimization -A comparison issue-

BACHIR BENHALA
LEAB, Faculty of Sciences
University of Moulay Ismaïl
B.P. 11201, Zitoune, Meknes, Morocco
b.benhala@fs-umi.ac.ma

*Abstract:* - Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Genetic Algorithm (GA) and Simulated Annealing (SA) are mostly used as metaheuristic for electronic circuit's performances optimization. Despite their different research techniques, these methods achieve the optimal solution for analog circuit design. The aim of this paper is to make a comparison between the performances reached by those four techniques in the optimal sizing of a CMOS second generation current conveyor (CCII) and an operational amplifier (Op-Amp). The highlighted results obtained by the used algorithms, will be compared in terms of optimum quality, convergence rate and the computing time.

*Key-Words:* - Swarm Intelligence, Evolutionary Algorithm, Simulated Annealing, Second Generation Current Conveyor, Operational Amplifier.

## 1 Introduction

Analogue circuit design deals with very complex nonlinear equations, obtaining optimal solution of these equations due to particular constraints in short time and acceptable error is an increasing challenge. It is then generally carried out thanks to the experiment and the intuition of the designer. Automatics procedures for the sizing of analog circuits are mostly suitable in order to design and simulate quickly a given electronic circuit performances and then reducing its time to market [1].

The best-known approaches in literature are based on fixed topologies and/or statistical techniques [2]. They are generally initialized with a "good" solution ("good" DC quiescent point) provided by the skilled analogue designer. The problem with these methods is that they are often very slow and they don't guarantee the convergence to a global optimum. The use of new methods is required. New heuristics based methods able to resolve optimization problems were then introduced [3], among which some are adaptable to many different problems referred to as metaheuristics. Their ability to optimize a problem from a minimum of information is compensated by the fact that they offer no guarantees about the optimality of the best found solution. Only an approximation of the global optimum is given at very reasonable times [4]. Starting from a current given point, in the research space, these methods generate new points by applying operators and by statistically moving toward more optimal places in the research space. Some (meta-)heuristics were proposed in the literature and used by analog designers, such as Tabu Search (TS) [5,6], Genetic Algorithms (GA) [7], Local search (LS) [8], Simulated Annealing (SA) [9], Ant Colony Optimization (ACO) [10-12], Wasp Nest (WN) [13], Bacterial Foraging Optimization (BFO) [14] and Particle Swarm Optimization (PSO) [15,16].

In this work, four different metaheuristic algorithms to solve typical analog circuit sizing problems will be used: Particle Swarm Optimization and Ant Colony Optimization which are derived from the Swarm Intelligence, Genetic Algorithm inspired by mechanisms from evolutionary algorithm and Simulated Annealing based on the physical phenomena. The aim is to compare these four techniques in terms of results quality and computing time for the resolution of analog sizing optimization problems. Two application examples are considered, a second generation current conveyor (CCII) and a two-stage CMOS operational amplifier (Op-Amp).

The rest of the paper is organized as follows. The Section 2 begins by a brief overview of the PSO, ACO, GA and SA techniques. The viability of the proposed algorithms, via some test problems, and the optimization results, obtained by each technique,

are presented and compared with each other in section 3 where is also presents the SPICE simulation results which show the validity of the obtained results. In section 4, we give a computing time and a checking of the convergence for the used algorithms. Finally, the major conclusions resulting from this work are presented and discussed.

# 2 An overview on the used metaheuristics

## 2.1 Swarm intelligence: PSO and ACO

The most used Swarm Intelligence (SI) techniques in the literature are Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) [17-19]. They are artificial intelligence techniques inspired by the collective behaviour of a group of animals such insects, birds and fish.

### 2.1.1 Particle Swarm Optimization:

The PSO is a stochastic global optimization method based on the social behaviour (bird flocking or fish schooling) and intelligence of swarm searching for the global optimal. This new method was developed by Edward and Kennedy in 1995 [17]. The PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating solution generations. However, unlike the GA, the PSO has no evolution operators such as crossover and mutation [20].

In the standard PSO, each particle I, is marked by two coordinates, a position-vector $x_{(i)}$ and a velocity a velocity-vector $v_{(i)}$. Each particle remembers its own best position and the global best position (best solution of the entire group).

The equations of velocity and position are illustrated by the following equations:

$$v_{(i+1)} = c\left\{\omega\, v_{(i)} + c_1 r_1\left(x_{lbest(i)} - x_{(i)}\right) + c_2 r_2\left(x_{gbest(i)} - x_{(i)}\right)\right\} \quad (1)$$

$$x_{(i+1)} = x_{(i)} + v_{(i)} \quad (2)$$

Where $v_{(i)}$ is the particle velocity and $x_{(i)}$, $x_{lbest(i)}$ and $x_{gbest(i)}$ are respectively the current particle, local best solution and global best solution at the $i^{th}$ generation, $r_1$ and $r_2$ are uniformly distributed in [0,1], $c_1$ and $c_2$ are the learning factors appointing two positive constants defined with an empirical way and according to the relation $c_1 + c_2 \leq 4$, $\omega$ is the inertia weight; it controls the impact of the previous velocity on the current one, and $c$ is a positive constant named constriction factor and used to control and constrict velocities.

During the iteration time t, the update of the velocity from the previous value to the new one is determined by eq. (1). The new position is then determined by the sum of the previous position and the new velocity by (2). Each particle, which flies over the solution space, remembers the encountered best solution. If this new solution has a cost less than the cost of the current global solution, then this best local solution replaces the best global solution. This process is then iterated many times.

The advantage of the PSO is its easiest implementation and its reduced number of parameters to adjust. The pseudo code of the PSO procedure can be presented as follows:

**Initialize** the size of the particle swarm and other parameters.
**Initialize** the positions and the velocities for all the particles randomly.
**Do**
  **For** each particle
    Calculate fitness value
    **If** the fitness value is better than the best fitness value (*pbest*) in history
      set current value as the new *pbest*
  **End**
    Choose the particle with the best fitness value of all the particles as the *gbest*
  **For** each particle
    Calculate particle velocity according to (1)
    Update particle position according to (2)
  **End**
**While** maximum iterations or minimum error criteria is not attained

Algorithm 1. Pseudo code of the PSO algorithm

### 2.1.2 Ant Colony Optimization:

Ant Colony Optimization technique is a meta-heuristic stochastic combinatorial computational discipline inspired by the behavior of ant colonies [21,22]. While walking, ants deposit pheromone on the ground marking a path that may be followed by other members of the colony. Shorter paths, as they accumulate pheromone faster than the longer ones, have a higher probability of being used by other ants, which again reinforce the pheromone on that path.

The ACO was first adapted to solve graph related problems, [23,24]. In such problems, ants randomly select the vertex to be visited. When an ant *k* is in

vertex *i*, the probability of going to vertex *j* is given [23,25] by:

$$p_{ij}^{k} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}.(\eta_{ij})^{\beta}}{\sum_{l \in J_i^k} (\tau_{il})^{\alpha}.(\eta_{il})^{\beta}} & if \quad j \in J_i^k \\ 0 & if \quad j \notin J_i^k \end{cases} \tag{3}$$

where $J_i^k$ is the set of neighbors of vertex *i* of the $k^{th}$ ant, $\tau_{ij}$ is the amount of pheromone trail on edge (*i,j*), $\alpha$ and $\beta$ are weightings that control the pheromone trail and the visibility value, i.e. $\eta_{ij}$, given by:

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{4}$$

$d_{ij}$ being the distance between vertices *i* and *j*.

The pheromone values are updated at each iteration by all the *m* ants that have built a solution in the iteration itself. The pheromone $\tau_{ij}$, which is associated with the edge joining vertices *i* and *j*, is updated as follows:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{5}$$

where $\rho$ is the pheromone evaporation rate, *m* is the number of ants, and $\Delta\tau_{ij}^{k}(t)$ is the quantity of pheromone laid on edge $(i, j)$ by the ant *k*:

Each ant *k* will randomly choose a path according to the probability given by expression (3), and form a directed graph while randomly generating a rate of pheromone at the formed graph edges. At each iteration, the path giving the minimum value of the Objective Function (OF) sees its pheromone rate increasing, in contrast with the other paths for which the pheromone rates are partially evaporated with respect to expression (5). The pseudo code of the ACO procedure can be presented as follows:

```
Random initialization of the pheromone value
Do
  For each iteration
    For each ant
      For each variable
        Compute of the probability P according to (3)
        Determine the Pmax
        Deduce the value of Vi
      End
     Compute OF
     End
   Deduce the best OF
   Update pheromone values according to (5)
  End
  Report the best solution
End
```

Algorithm 2. Pseudo code of the ACO algorithm

## 2.2 Genetic Algorithm

*Darwin*'s theory about evolution inspired Holland and Goldberg [26,27]; they mimic natural evolution processes to find a solution to a problem: combination of selection, recombination and mutation. They form a subclass of evolutionary algorithms. Figure 1 illustrates its basic principle.

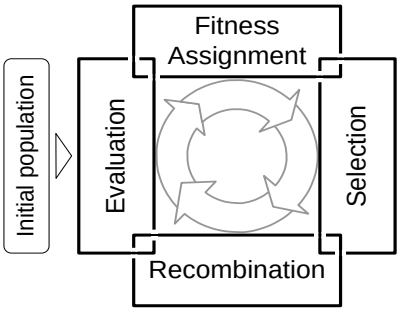



Fig. 1 The basic cycle of evolutionary algorithms

where *Evaluation* consists of computing the objective values of the solution candidates. *Fitness Assignment* uses the objective values to determine fitness values. *Selection* chooses the fittest individuals for reproduction, and *Reproduction* creates new individuals from the mating pool by crossover and mutation [28].

The pseudo code of a basic GA is given in Algorithm 3.

```
    InitPopulation ( ) // random initialization of the
population
    max_fitness := 0
    For each member chromosome
        fitness := Fitness_Evaluation (chomosome)
            If fitness > max_fitness
                max_fitness := fitness
                fittest_solution = chromosome
            End
    End
    while generation < max_generations
            offspring := Selection (parents) // using
roulette-wheel selection
            fitness := Fitness_Evaluation (offspring)
            If fitness > max_fitness
                max_fitness := fitness
                fittest_solution = offspring
            End
    save fittest_solution
End
```

Algorithm 3. Pseudo code of the GA algorithm

### 2.3 Simulated Annealing

The method of simulated annealing (SA) was first introduced by Kirkpatrick [29]. This technique simulates the annealing process in which a solid, in a heat bath, is heated above its melting temperature and then progressively, the temperature is lowered slowly to produce the crystalline lattice, which reduces its energy probability distribution. This crystalline lattice is an attractive example of nature finding of an optimal configuration [30].

Generally, SA exploits the Metropolis Algorithm [31] which provides the criterion for acceptance of a solution 'x' constructed by disrupting the current solution x. This process gives the possibility of moving away occasionally of a local minimum to allow a widening of the research field of the ideal solution. There are several distinct steps that the SA process has to go through as the temperature is condensed and randomness is applied to the input values.

The pseudo code of the SA procedure is as follows:

Setup the SA.
Choose a random solution (x) which takes the minimum solution $x_{min}$ ($x=x_{min}$)
Evaluate the cost function f (x) equal to f ($x_{min}$)
Initialize the temperature T
**Repeat**
  **Repeat**
    Generate a neighbor X' perturbing the solution X.
      Acceptance with the criterion of Metropolis
      **if** f (x') ≤ f (x)
        f (x) = f (x')
        x=x'
      **if** not
        Calculate Δ f = f (x') - f (x)
        Calculate the probability p = exp (- Δ f / T)
        Choose a random solution R between [0, 1]
        **if** R ≤ p
          f (x) = f (x')
          x= x'
        **end if**
        $f_{min}$ = f (x) and $x_{min}$ = x
      **end if**
  **Until** thermodynamic equilibrium reaches
T: = to decrease temperature
**Until** maximum iterations or minimum error criteria is attained

Algorithm 4. Pseudo code of the SA algorithm

## 3 Test and application examples

Under similar conditions, the four metaheuristics were applied to the optimal design of the aforementioned two CMOS circuits and two test functions.
The proposed MATLAB-implemented algorithms parameters are given in Table 1 with a generation algorithm of 200 with Pentium (R) Dual-Core CPU, T4500 @2.3 GHz.

Table 1. Parameters of algorithms

| | | |
|---|---|---|
| **PSO** | Swarm size | 100 |
| | inertia weight (w) | 0.9 |
| | Particle Velocity (c1) | 1 |
| | Particle Position (c2) | 3 |
| **ACO** | Number of Ants | 100 |
| | Evaporation rate (ρ) | 0.1 |
| | Quantity of deposit pheromone (Q) | 0.2 |
| | Pheromone Factor (α) | 1 |
| | Heuristics Factor (β) | 1 |
| **GA** | Population size | 100 |
| | Crossover Probability | 0.9 |
| | Mutation Probability | 1e-4 |
| **SA** | Initial temperature (T) | 100 |
| | Annealing rate | 0.9 |
| | Final temperature | 1e-8 |

### 3.1 The test examples

In order to check the performances of the proposed algorithm, two classical objectives test functions [28,32] were used; their expressions are given in Table 2.

Table 2. The test functions

| | Variable Bounds | Objective Functions |
|---|---|---|
| F1 | $-4 \le x, y \le 4$ | $f(x,y)=(1.5-x+xy)^2+\left(2.25-x+xy^2\right)^2+\left(2.625-x+xy^3\right)^2$ |
| F2 | $0 \le x, y \le 10$ | $f(x,y)=x\sin(4x)+1.1y\sin(2y)$ |

In Table 3 we present the obtained minimas by all the studied algorithm as well as the theoretical know results.

Table 3. Comparison algorithm results and theoretical results

| | theoretical results | PSO | ACO | GA | SA |
|---|---|---|---|---|---|
| **F1** | 0.00 | 0.00 | 0.00 | 0.00 | 1.1581e-05 |
| **F2** | -18.5547 | -18.5547 | -18.5547 | -18.5517 | -18.5516 |

We can notice that, the good agreement between theoretical results and those obtained using the proposed algorithms. The techniques ACO and PSO provide the global minima, whereas the GA and SA give closer but acceptable results.

### 3.2 The application examples

In the previous work, we have described our proposed algorithms and shown their performance on a set of well-known test functions. In this part, the abovementioned algorithms were used to optimize performances of the following analog circuits and performances under consideration are:

- The X-port parasitic resistance of a CMOS second generation current conveyor (CCII).
- The high cut-off frequency of a CMOS second generation current conveyor (CCII).
- The Open-loop voltage gain and the Common Mode Rejection Ratio, the Die Area and the Power dissipation of a two-stage CMOS Op-Amp.

#### 3.2.1 Performance optimization of a CMOS CCII

The four considered techniques, i.e. PSO, ACO, GA and SA, were used to optimize performances of a CCII circuit that is shown on Figure 2.

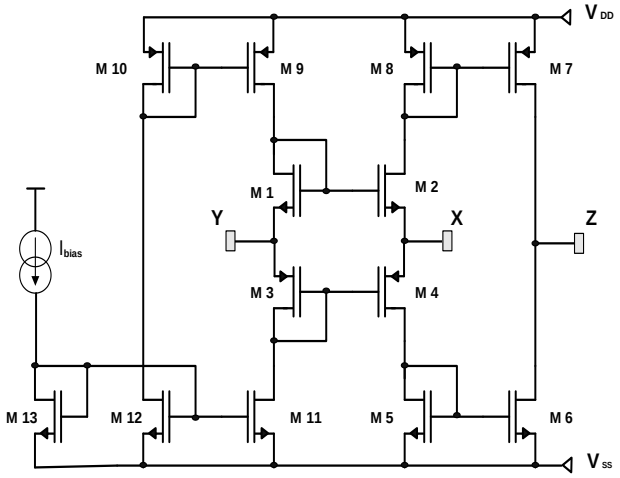



Fig. 2 A second generation current conveyor (CCII)

#### 3.2.1.1 Objective functions and constraints

The objective functions to be optimized are [33,34]:

- $R_X$: the X-port input parasitic resistance to be minimized.
- *fci*: the current high cut off frequency to be maximized.

All the CCII transistors must operate in the saturation mode. Saturation constraints of each MOSFET were determined [35].

The geometric variables to be considered for the CMOS CCII performances optimization are the MOS transistors sizes: channels lengths (LN, and LP) and gates widths (WN and WP) while respecting the saturation conditions of the MOS transistors. The used supply voltages are $V_{DD}/V_{SS}$=2.5V/-2.5 V and a bias current, $I_{bias}$=100 µA.

#### 3.2.1.2 Optimization results

Tables 4 and 5 give optimal results obtained by using the PSO, the ACO, the GA and the SA algorithms for the parameters and the circuit's performances for CCII.

Table 4. Optimization and simulation results for $Rx_{min}$

| | LN (µm) | WN (µm) | LP (µm) | WP (µm) | $Rx_{min}$ (Ω) | |
|---|---|---|---|---|---|---|
| | | | | | Opt. | Sim. |
| **PSO** | 0.59 | 18.74 | 0.35 | 30.00 | **464** | 477 |
| **ACO** | 0.55 | 20.36 | 0.36 | 30.00 | **441** | 451 |
| **GA** | 0.50 | 13.86 | 0.35 | 27.32 | **529** | 512 |
| **SA** | 0.55 | 17.49 | 0.35 | 30.00 | **473** | 475 |

Table 5. Optimization and simulation results for $fci_{max}$

| | LN (µm) | WN (µm) | LP (µm) | WP (µm) | $fci_{max}$ (GHz) | |
|---|---|---|---|---|---|---|
| | | | | | Opt. | Sim. |
| **PSO** | 0.54 | 05.80 | 0.35 | 10.05 | **1.751** | 1.784 |
| **ACO** | 0.56 | 05.39 | 0.35 | 09.49 | **1.791** | 1.887 |
| **GA** | 0.58 | 5.96 | 0.36 | 10.56 | **1.576** | 1.621 |
| **SA** | 0.55 | 8.16 | 0.35 | 14.13 | **1.422** | 1.485 |

Figure 3 and 4 show the simulation results (*Rx* and *fci*) corresponding to the application of the PSO, ACO, GA and SA techniques for CCII.

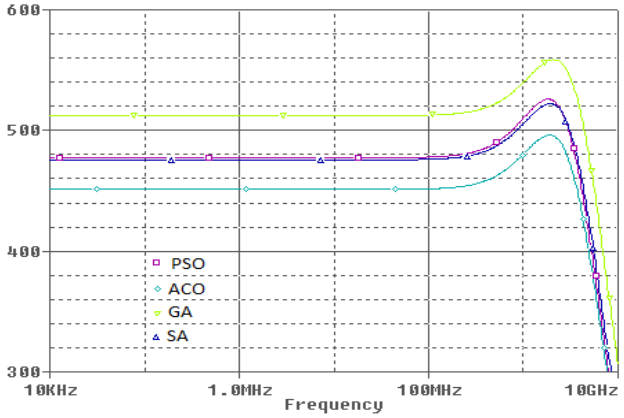



Fig. 3 Rx-pole resistance (Ω) *vs.* frequency (Hz)

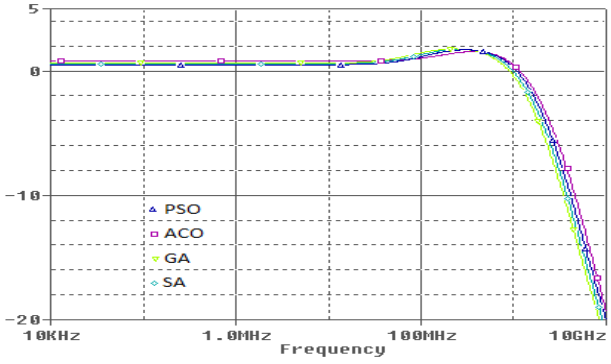



Fig. 4 Current gain (dB) *vs.* frequency (Hz)

### 3.2.2 Optimizing performances of a CMOS operational amplifier

The schematic of the two stage CMOS operational amplifier is shown in Figure 5. Performances of the op-amp are evaluated via several parameters [10]:

- The Open-loop voltage gain *Av* to be maximized.
- The Common Mode Rejection Ratio *CMRR* to be maximized.
- The Die Area *A* to be minimized.
- The Power dissipation *P* to be minimized.

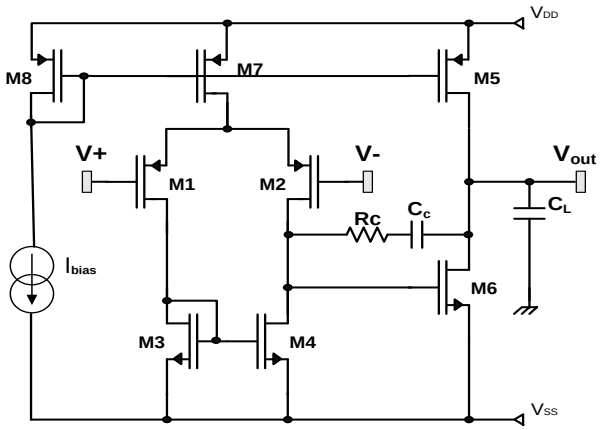



Fig. 5 A two stage CMOS operational amplifier

Determining the optimal dimensions of the transistors for a specific design involves a tradeoff among all these performance measures.
Ensuring the saturation of each transistor of the circuit constitutes the constraints of the problem [10].

The considered optimization problem is a multi-objective one that consists of maximizing two objective functions such as the *Av*, *CMRR* and minimizing the two other performances; the die area and the consumed power. In this work we focus on the mono-objective optimization. For this we will use the weighted cost functions technique [28]. It consists of weighting each function and transforming the set of objectives into an equivalent cost function (OF). Thus, OF can be expressed as follows:

$$OF = \alpha_1 A_v + \alpha_2 P + \alpha_3 A + \alpha_4 CMRR \qquad (6)$$

where $\alpha_{1-4}$ are normalization coefficients.

The compensation resistor ($R_C$=800Ω), the compensation capacitor ($C_C$=3pF) and the capacitive load ($C_L$=10pF) are considered as fixed parameters.
The transistors M1-M8 have a same channel length L but several gates widths W1-W8. Those parameters and the bias current $I_{bias}$ were optimized by the four algorithms. A Table 6 shows the obtained optimal values and the Table 7 gives the associate performances and simulations results.

Table 6. Optimization results

| | $W_{1,2}$ (µm) | $W_{3,4}$ (µm) | $W_5$ (µm) | $W_6$ (µm) | $W_7$ (µm) | $W_8$ (µm) | L (µm) | Io (µA) |
|---|---|---|---|---|---|---|---|---|
| **PSO** | 213.59 | 255.01 | 56.37 | 477.99 | 52.83 | 09.24 | 0.35 | 10.24 |
| **ACO** | 215.02 | 260.96 | 57.82 | 459.52 | 50.91 | 09.65 | 0.35 | 10.00 |
| **GA** | 211.49 | 260.09 | 79.42 | 404.09 | 61.69 | 09.52 | 0.35 | 10.16 |
| **SA** | 216.38 | 260.05 | 59.03 | 470.11 | 53.35 | 10.00 | 0.35 | 10.00 |

Table 7. Performance and simulation results for *Av*, *CMRR*, *A* and *P*

| | | Av (dB) | CMRR (dB) | A ($\mu m^2$) | P (mW) |
|---|---|---|---|---|---|
| **PSO** | Opt. | **103.51** | **110.57** | **5368** | **1.3** |
| | Sim. | 095.43 | 101.43 | --- | 1.8 |
| **ACO** | Opt. | **104.13** | **113.91** | **5354** | **1.2** |
| | Sim. | 096.74 | 105.29 | --- | 1.7 |
| **GA** | Opt. | **099.76** | **105.23** | **5242** | **1.6** |
| | Sim. | 092.86 | 096.58 | --- | 2.4 |
| **SA** | Opt. | **104.50** | **108.35** | **5408** | **1.2** |
| | Sim. | 097.31 | 099.43 | --- | 1.8 |

Figure 6 and 7 show the simulation results (*Av* and *CMRR*) corresponding to the application of the PSO, ACO, GA and SA techniques for Op-Amp.

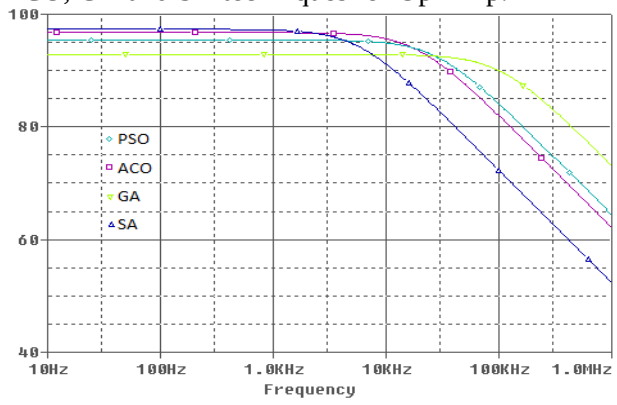



Fig. 6 Gain (dB) *vs.* frequency (Hz)

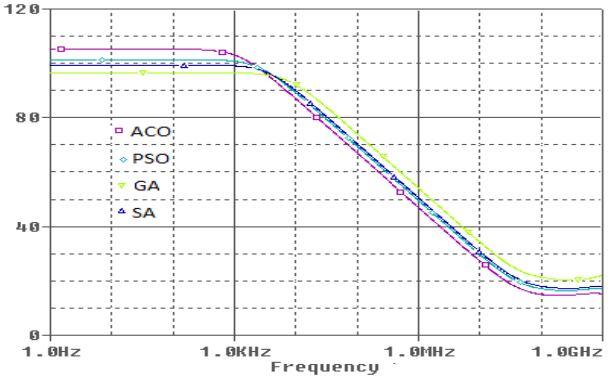



Fig. 7 *CMRR* (dB) *vs.* frequency (Hz)

# 4 Computing time and Convergence checking

## 4.1 Computing time

Table 8 shows a comparison between average computing times for 100 runs of each algorithm. Figure 8 shows a recapitulation of the corresponding computing times.

Table 8. Average computing time (seconds) for the test functions and the studied circuits

| Circuits and test functions | PSO | ACO | GA | SA |
|---|---|---|---|---|
| F1 | 0.1275 | 2.1392 | 0.1467 | 0.6753 |
| F2 | 0.1628 | 2.4263 | 0.1563 | 0.9811 |
| Rx | 2.5876 | 7.2716 | 2.8982 | 4.0234 |
| fci | 2.4331 | 9.3154 | 2.5652 | 3.6842 |
| Ampli-Op | 4.3170 | 22.183 | 2.4972 | 3.1442 |

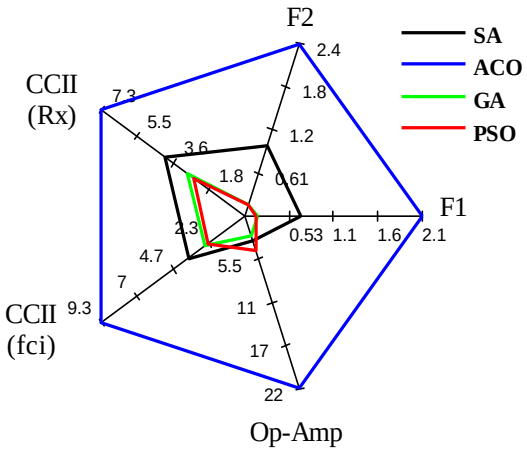
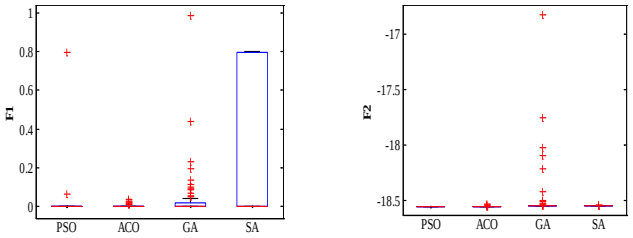



Fig. 8 Execution time (seconds) for the five objective problems

We clearly notice that the ACO algorithm is the slower compared to other algorithms.

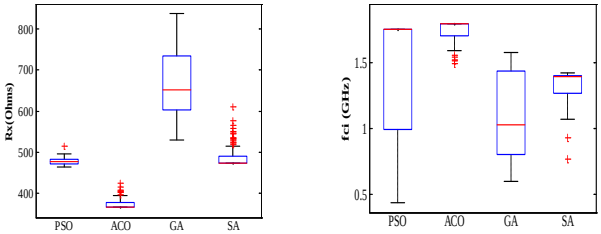
## 4.2 Convergence checking

In order to check the convergence rate of the proposed algorithms, a robustness test was performed. i.e. the algorithms are applied a hundred times for optimizing all objectives ( *F1, F2, Rx, fc, Av, CMRR, P* and *S*). In Figures 9, 10 and 11 we present the Box Plots of each algorithm for each objective considered function.



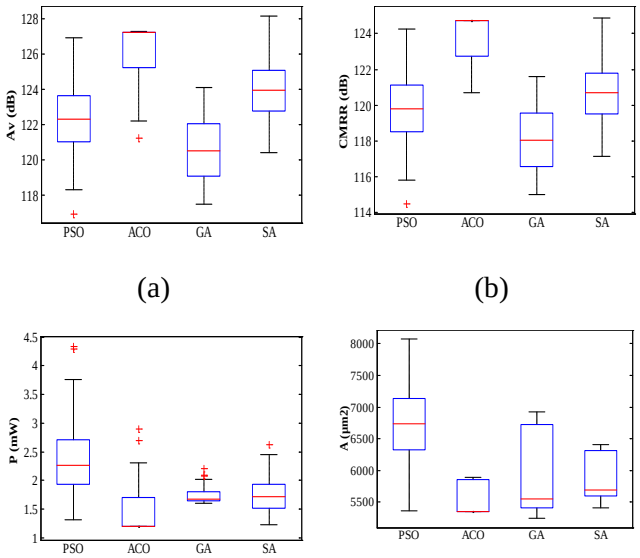

(a) (b)

Fig. 9 Box plot for the convergence rate: (a) for F1 and (b) for F2




(a) (b)

Fig. 10 Box plot for the convergence rate: (a) for *Rx* and (b) for *fci*




(a) (b)

(c) (d)

Fig. 11 Box plot for the convergence rate: (a) for *Av*, (b) for *CMRR*, (c) for *P* and (d) for *A*

We can be easily noticed the good convergence ratio, despite the probabilistic aspect of the algorithms. We note that the spacing between the optimal values given by PSO and GA algorithms is larger than those of ACO and SA techniques which therefore show a good degree of dispersion. In the other hand, the PSO technique has the most outlier

values and the GA presents the best symmetry of the values. We can, also, notice that the robustness of the ACO algorithm is better than the robustness of the PSO, GA and SA algorithms.

# 5 Conclusion

The presented work presents a comparison between four metaheuristic techniques: Particle Swarm Optimization, Ant Colony Optimization, Genetic Algorithm and Simulated Annealing. Each technique was first validated through mathematical test functions and used for the optimal sizing of two analog circuits; a CMOS second generation current conveyor and an operational amplifier. The considered algorithms achieve the optimal solution; the choice between these algorithms will depend on the desiderata of the designer. However, our results suggest that a hybrid algorithm consisting of at least two techniques, one taking care of optimum quality, the other taking care of running time, is a promising research direction.

*References:*


[1] C. Toumazou, and F. J. Lidgey, In Analog IC Design: The current mode approach, *Haigh, D. G. (Ed.), IEEE circuit and systems* series 2. 1993.

[2] F. Medeiro, R. Rodríguez-Macías, F.V. Fernández, R. Domínguez-Astro, J.L. Huertas, and A. Rodríguez-Vázquez, Global design of analog cells using statistical optimization techniques, *Analog Integrated Circuits and Signal Processing*, Vol. 6, N° 3, 1994, pp. 179–195.

[3] C.R. Reeves, Modern heuristic techniques for combinatorial problems, *(Ed.) Blackwell Scientific Publications*, Oxford, 1993.

[4] I.H. Osman and J.P. Kelly, Meta-heuristics: theory and applications, *Kluwers Academic Publishers*, Boston, 1996.

[5] F. Glover, Tabu search-part I, *ORSA Journal on computing*, Vol. 1, N° 3, 1989, pp. 190–206.

[6] F. Glover, Tabu search-part II, *ORSA Journal on computing*, Vol. 2, N° 1, 1990, pp. 4–32.

[7] J. B. Grimbleby, Automatic analogue circuit synthesis using genetic algorithms, *IEE Proceedings-Circuits, Devices and Systems*, Vol. 147, N° 6, 2000, pp. 319–323.

[8] E. Aarts and K. Lenstra, Local search in combinatorial optimization, *Princeton: Princeton University Press*, 2003.

[9] M. Fakhfakh, M. Bougharriou, A. Sallem and M. Loulou, Design of Low Noise Amplifiers through Flow-Graphs and their Optimization by the Simulated Annealing Technique, *Book: Advances in Monolithic Microwave Integrated Circuits for Wireless Systems: Modeling and Design Technologies, IGI global*, pp 89-103, 2012.

[10] B. Benhala, A. Ahaitouf, M. Kotti, M. Fakhfakh, B. Benlahbib, A. Mecheqrane, M. Loulou, F. Abdi and E. Abarkane, Application of the ACO Technique to the Optimization of Analog Circuit Performances, *Chapter 9, Book: Analog Circuits: Applications, Design and Performance, Ed., Dr. Tlelo-Cuautle, NOVA Science Publishers*, pp. 235–255. 2011.

[11] B. Benhala, A. Ahaitouf, A. Mechaqrane, B. Benlahbib, F. Abdi, E. Abarkan and M. Fakhfakh, Sizing of current conveyors by means of an ant colony optimization technique, *The IEEE International Conference on Multimedia Computing and Systems (ICMCS'11)*, 2011, pp. 899–904, Ouarzazate, Morocco.

[12] B. Benhala, A. Ahaitouf, A. Mechaqrane and B. Benlahbib, Multiobjective optimization of second generation current conveyors by the ACO technique, *The International Conference on Multimedia Computing and Systems (ICMCS'12)*, 2012, pp. 1147–1151, Tangier, Morocco.

[13] F. T. S. Chan and M. K. Tiwari, Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, *I-Tech Publishing*. December 2007.

[14] A. Chatterjee, M. Fakhfakh and P. Siarry, Design of Second Generation Current Conveyors Employing Bacterial Foraging Optimization, *Microelectronics Journal, Elsevier*, Vol. 41, 2010. pp. 616–626.

[15] J. Kennedy and R. C. Eberhart, Particle swarm optimization, *The IEEE International conference on neural networks, WA*, Australia, November 27–December 1, 1995.

[16] M. Fakhfakh, Y. Cooren, A. Sallem, M. Loulou, and P. Siarry Analog Circuit Design Optimization through the Particle Swarm Optimization Technique, *Journal of Analog Integrated Circuits & Signal Processing, Springer*, Vol. 63, N° 1, 2010. pp. 71–82.

[17] J. Kennedy, R. C. Eberhart and Y. Shi, Swarm Intelligence, *Morgan Kaufmann*, 2001.

[18] A. P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, *Wiley & Sons*, 2006.

[19] E. Bonabeau, M. Dorigo and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, *Oxford University Press US*, 1999.
[20] http://www.swarmintelligence.org/
[21] E. Bonabeau, M. Dorigo and G. Theraulaz, Inspiration for optimization from social insect behavior, *Nature,* Vol. 406, 2000, pp. 39–42.
[22] S. D. Shtovba, Ant Algorithms: Theory and Applications, *Programming and Computer Software,* Vol. 31, N° 4, 2005, pp. 167–178.
[23] Y. Jinhui, S. Xiaohu, M. Maurizio and L. Yanchun, An ant colony optimization method for generalized TSP problem, *Progress in Natural Science*, Vol. 18, 2008, pp 1417–1422.
[24] B. Yu, Z. Yang and B. Yao, An improved ant colony optimization for vehicle routing problem, *European Journal of Operational Research*, Vol. 196, , 2009, pp. 171–176.
[25] M. Dorigo and S. Krzysztof, An Introduction to Ant Colony Optimization, *a chapter in Approximation Algorithms and Metaheuristics, a book edited by T. F. Gonzalez,* 2006.
[26] J. H. Holland, Adaptation in Natural and Artificial Systems, *Ann Arbor: University of Michigan Press*. 1975.
[27] A.E. Eiben and J.E. Smith, Introduction to evolutionary computing, *Springer, ISBN: 978-3-540-40184-1*. 2007.
[28] R.L. Haupt and S.E. Haupt, Practical Genetic Algorithms, *John Wiley & Sons, ISBN 0-471-45565-2*. 2004.
[29] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, Optimization by simulated annealing, *Journal of Science*, Vol. 220, N° 4598, 1983, pp. 671-680.
[30] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, Equation of State Calculations by Fast Computing Machines, *J. Chem. Phys.*, Vol. 21, N° 6, 1953, pp.1087–1092,
[31] J. P. Courat, G. Raynaud, I. Mrad and P. Siarry, Electronic component model minimization based on Log Simulated Annealing, *IEEE Transactions on Circuits and Systems*, Vol. 41, 1994, pp. 790–795.
[32] http://www.optima.amp.i.kyotou.ac.jp/member/student/hedar/Hedar_files/TestGO_files
[33] A. Sedra and K. C. Smith, A second generation current conveyor and its applications. *IEEE Transactions on Circuit Theory*, Vol. 17, 1970, pp. 132–134.
[34] M. Fakhfakh and M. Loulou, Live demonstration: CASCADES. 1: A flow-graph-based symbolic analyzer, *the IEEE International symposium on circuits and systems*, 2010.
[35] B. Benhala and O. Bouattane, GA and ACO techniques for the analog circuits design optimization, *Journal of Theoretical and Applied Information Technology*, Vol. 64, N° 2, 2014, pp. 413-419.