A new heuristic approach for non-convex optimization problems

R. Toscano[1], P. Lyonnet

Université de Lyon
Laboratoire de Tribologie et de Dynamique des Systèmes CNRS UMR5513 ECL/ENISE
58 rue Jean Parot 42023 Saint-Etienne cedex 2

Abstract. In this work a new optimization method, called the heuristic Kalman algorithm (HKA), is presented. This new algorithm is proposed as an alternative approach for solving continuous, non-convex optimization problems. The principle of HKA is to explicitly consider the optimization problem as a measurement process designed to give an estimate of the optimum. A specific procedure, based on the Kalman estimator, was developed to improve the quality of the estimate obtained through the measurement process. The main advantage of HKA, compared to other metaheuristics, lies in the small number of parameters that need to be set by the user. Further, it is shown that HKA converges almost surely to a near-optimal solution. The efficiency of HKA was evaluated in detail using several non-convex test problems, both in the unconstrained and constrained cases. The results were then compared to those obtained via other metaheuristics. The numerical experiments show that HKA is a promising approach for solving non-convex optimization problems, particularly in terms of computation time and success ratio.

Keywords: heuristic Kalman algorithm, non-convex optimization, metaheuristic, objective function, almost sure convergence.

1 Introduction

In all areas of engineering, physical and social sciences, problems involving the optimization of some objective function are encountered. Usually, the problem that needs to be solved can be formulated precisely, but is often difficult or impossible to solve either analytically or through conventional numerical procedures. This is the case when the problem is non-convex and thus inherently nonlinear and multimodal. In fact, it is now well-established that the convexity of optimization problems determines whether they can be efficiently solved or not [24]. Today, very efficient algorithms for solving convex problems exist [2], but the problem of non-convex optimization remains largely open, despite an enormous amount of effort that has been devoted to its resolution.

[1]E-mail address: toscano@enise.fr, Tel.:+33 477 43 84 84; Fax: +33 477 43 84 99

Evolutionary approaches as well as other-population based methods handle non-convex optimization problems well [18, 10, 9, 29, 11, 28, 14, 19]. This is the reason behind the success and the broad diffusion of evolutionary optimization methods such as the well-known genetic algorithm (GA) [16, 13]. The main characteristic of these kinds of approaches, also called metaheuristics, is the use of a stochastic mechanism to find a solution. From a general point of view, the use of a stochastic search procedure appears essential for finding a promising solution.

Following this kind of approach, we propose a new optimization method, called the heuristic Kalman algorithm (HKA). Our approach falls in the category of the so-called "population-based stochastic optimization techniques". However, its search heuristic is entirely different from other known stochastic algorithms. Indeed, HKA explicitly considers the optimization problem as a measurement process designed to give an estimate of the optimum. A specific procedure, based on the Kalman estimator, was developed to improve the quality of the estimate obtained through the measurement process. The main advantage of HKA compared to other metaheuristics, lies in the small number of parameters that need to be set by the user (only three). This property makes the algorithm easy to use for non-specialists.

The efficiency of HKA was evaluated in detail using several non-convex test problems, both in unconstrained and constrained cases. The results were then compared to those obtained via other metaheuristics. The numerical experiments show that the HKA has promising potential for solving non-convex optimization problems, notably in terms of the computation time and success ratio.

The paper is organized as follows. In section 2, HKA is presented and its convergence properties are given. In Section 3, various numerical experiments are conducted to evaluate the efficiency of HKA in solving non-convex optimization problems. Finally, section 4 concludes the paper.

2 The heuristic Kalman algorithm (HKA)

Consider the general system presented in figure 1, which produces an output in response to a given input. In addition, this system has some tuning parameters that can modify its behavior. By behavior, we mean the relationship existing between the inputs and the outputs.

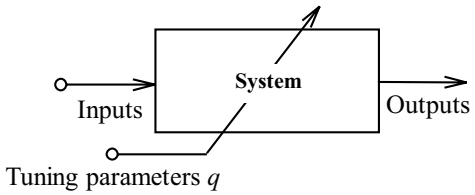

Figure 1: An optimization problem.

The problem is how to tune these parameters so that the system behaves well. Usually, the desired behavior can be formulated via an objective function that depends on the tuning parameters $J(q)$, which needs to be maximized or minimized with respect to q.

More formally, the problem to be solved can be formulated as follows: find the optimal tuning parameters q_{opt}, solution of the following problem:

$$\begin{cases} q_{opt} = \arg\min_{q \in \mathcal{D}} J(q) \\ \mathcal{D} = \{q : g_i(q) \leqslant 0,\ i = 1, \cdots, n_q\} \end{cases} \tag{1}$$

where $J : \mathbf{R}^{n_q} \to \mathbf{R}$ is a function for which the minimum ensures that the system behaves as desired and g_i are some constraints on the parameters q. The objective is to find the tuning vector of parameters q_{opt} that belong to the set of admissible solutions $\mathcal{D}$ that minimize the cost function J. Unfortunately, there are several obstacles in solving this kind of problem. The main obstacle is that most optimization problems are NP-hard [12]. Therefore, the known theoretical methods cannot be applied except possibly for some small sized problems. Another difficulty is that the cost function may be not differentiable and/or multimodal. Therefore, methods that require derivatives of the cost function cannot be

used. Another obstacle is when the cost function cannot be expressed in an analytic form; in this case, the cost function can be only evaluated through simulations.

In these situations, heuristic approaches seem to be the only way to solve optimization problems. By a heuristic approach, we mean a computational method employing experimentations, evaluations and trial-and-error procedures to obtain an approximate solution for computationally difficult problems. This type of approach was used to develop HKA and is described in the next section.

2.1 Principle of the algorithm

The main idea of HKA is to generate, via experiments (measurements), a new point which is hopefully closer to the optimum than the preceding point. This process of estimation is repeated until no further improvements can be made. More precisely, we adopt the principle depicted in figure 2. The proposed procedure is iterative, and we denote by k the k^{th} iteration of the algorithm. We have a random generator of a probability density function (pdf) $f(q)$, which produces, at each iteration, a collection of N vectors that are distributed around a given mean vector m_k with a given variance-covariance matrix Σ_k. This collection can be written as follows:

$$\mathbf{q}(k) = \left\{ q_k^1,\ q_k^2,\ \cdots,\ q_k^N \right\}, \tag{2}$$

where q_k^i is the i^{th} vector generated at iteration k: $q_k^i = [q_{1,k}^i \cdots q_{n,k}^i]^T$, and $q_{l,k}^i$ is the l^{th} component of q_k^i $(l = 1, \cdots, n)$. This random generator is applied to the cost function J. Without loss of generality, we assume that the vectors are ordered by their increasing cost function, i.e.

$$J(q_k^1) < J(q_k^2) < \cdots < J(q_k^N). \tag{3}$$

The principle of the algorithm is to modify the mean vector and the variance-covariance matrix of the random generator until the minimum of the cost function is reached.

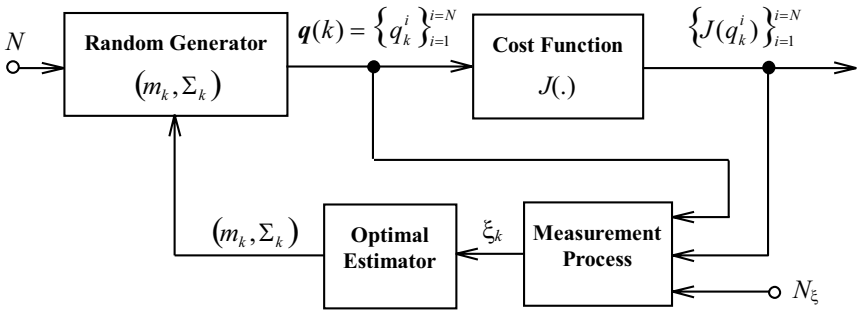

Figure 2: Principle of the algorithm

More precisely, let N_ξ be the number of best samples under consideration, such that $J(q_k^{N_\xi}) < J(q_k^i)$ for all $i > N_\xi$. Note that the best samples are those of sequence (2) which have the smallest cost function. The objective is then to generate, from the best samples, a new random distribution which approaches, on average, the minimum of the cost function J with decreasing variance.

To this end, a measurement procedure followed by an optimal estimator of the parameters of the random generator is introduced. The measurement process consists in computing the average of the candidates that are the most representative of the optimum. For the k^{th} iteration, the measurement, denoted ξ_k, is defined as follows:

$$\xi_k = \frac{1}{N_\xi} \sum_{i=1}^{N_\xi} q_k^i, \tag{4}$$

where N_ξ is the number of considered candidates. We can consider that this measurement gives a perturbed knowledge about the optimum, i.e.

$$\xi_k = q_{opt} + v_k, \tag{5}$$

where v_k is an unknown disturbance, centered on q_{opt}, and acting on the measurement process. The components of v_k are assumed to be independent and follow a centered normal distribution. Since the covariance between two independent variables is zero, all non-diagonal elements of variance-covariance matrix are zero. Thus, the variance-covariance matrix of v_k is given by:

$$V_k = \frac{1}{N_\xi} \begin{bmatrix} \sum_{i=1}^{N_\xi} (q_{1,k}^i - \xi_{1,k})^2 & & 0 \\ & \ddots & \\ 0 & & \sum_{i=1}^{N_\xi} (q_{n,k}^i - \xi_{n,k})^2 \end{bmatrix}. \tag{6}$$

In other words, the diagonal of V_k (denoted $\mathrm{vec}^{\mathrm{d}}(V_k)$) represents the variance vector. Note that this variance vector can be used to measure our ignorance about q_{opt}. In these conditions, the Kalman filter can be used to make a so-called "a posteriori" estimate of the optimum, i.e. it accounts for both the measurement and the confidence placed in it. As seen, this confidence can be quantified by (6). Roughly speaking, a Kalman filter is an optimal recursive data-processing algorithm [21]. Optimality must be understood as the best estimate that can be made based on the model used for the measurement process as well as the data used to compute this estimate.

Our objective is to design an optimal estimator which combines a prior estimation of q_{opt} and the measurement of ξ_k, so that the resulting posterior estimate is optimal in a sense which will be defined below. In the Kalman framework, this kind of estimator takes the following form:

$$\hat{q}_k^+ = L_k' \hat{q}_k^- + L_k \xi_k, \tag{7}$$

where $\hat{q}_k^-$ represents the prior estimation, i.e. before measurement, $\hat{q}_k^+$ is the posterior estimation i.e. after measurement, L_k' and L_k are unknown matrices which have to be determined to ensure optimal estimation. Here optimality is reached when the expectation of the posterior estimation error is zero and its variance is minimal. This can be expressed as follows:

$$\begin{cases} (L_k',\, L_k) = \arg \min\limits_{L_k',\, L_k} \mathbf{E}[\tilde{q}_k^{+T} \tilde{q}_k^+] \\ \mathbf{E}[\tilde{q}_k^+] = 0 \end{cases}, \tag{8}$$

where $\mathbf{E}$ is the expectation operator and $\tilde{q}_k^+$ represents the posterior estimation error at

iteration k. We define the posterior estimation error and its variance-covariance matrix as

$$\tilde{q}_k^+ = q_{opt} - \hat{q}_k^+, \quad P_k^+ = \mathbf{E}[\tilde{q}_k^+ \tilde{q}_k^{+T}]. \tag{9}$$

Similarly, we define the prior estimation error and its variance-covariance matrix as

$$\tilde{q}_k^- = q_{opt} - \hat{q}_k^-, \quad P_k^- = \mathbf{E}[\tilde{q}_k^- \tilde{q}_k^{-T}]. \tag{10}$$

Under the assumption that $\mathbf{E}[\tilde{q}_k^-] = 0$, it can be easily established that the satisfaction of the condition $\mathbf{E}[\tilde{q}_k^+] = 0$ requires

$$L_k' = I - L_k, \tag{11}$$

where I is the identity matrix. Then, putting this expression into equation (7) gives:

$$\hat{q}_k^+ = \hat{q}_k^- + L_k(\xi_k - \hat{q}_k^-). \tag{12}$$

The objective is now to determine L_k in such a way that the variance of the posterior estimation error is minimized. Noting that $\text{trace}(P_k^+) = \mathbf{E}[\tilde{q}_k^{+T} \tilde{q}_k^+]$, the minimization of the variance of q_k^+ is accomplished by minimizing the trace of P_k^+ with respect to L_k. Standard calculus, similar to the one used for the derivation of the Kalman filter, yields [21]

$$L_k = P_k^-(P_k^- + V_k)^{-1}, \quad P_k^+ = (I - L_k)P_k^-. \tag{13}$$

Finally, for the k^{th} iteration, the HKA algorithm utilizes the relations (4), (6), (12), (13) and the next iteration is initialized with $m_k = q_k^+$, $\Sigma_k = P_k^+$ and $k = k + 1$. Practical considerations for efficient use of HKA are given in the next section.

2.2 Some practical considerations

The expression used for computing P_k^+ (see 13) generally leads to a decrease in the variance of the Gaussian distribution that is too fast, which results in a premature convergence of the algorithm. This difficulty can be tackled by introducing a slowdown factor in S_k^+ (the posterior standard deviation vector of the Gaussian generator) and adjusting it according

to the dispersion of the best candidates considered for the estimation of q_{opt}. This can be done as follows:

$$S_k^+ = S_k^- + a_k(W_k - S_k^-), \quad \text{with:} \quad \begin{cases} a_k = \dfrac{\alpha \min\left(1, \left(\frac{1}{nq}\sum_{i=1}^{nq}\sqrt{v_{i,k}}\right)^2\right)}{\min\left(1, \left(\frac{1}{nq}\sum_{i=1}^{nq}\sqrt{v_{i,k}}\right)^2\right) + \max_{1\leqslant i\leqslant nq}(w_{i,k})} \\ S_k^- = \left(\mathrm{vec}^{\mathrm{d}}(P_k^-)\right)^{1/2},\ W_k = \left(\mathrm{vec}^{\mathrm{d}}(P_k^+)\right)^{1/2} \end{cases}, \tag{14}$$

where S_k^- and S_k^+ are, respectively, the prior and the posterior standard deviation vectors, a_k is the slowdown factor, $\alpha \in (0,\, 1]$ the slowdown coefficient, and $v_{i,k}$ represents the i^{th} component of the variance vector $\mathrm{vec}^{\mathrm{d}}(V_k)$ defined in (6), $w_{i,k}$ is the i^{th} component of the vector W_k, and $\mathrm{vec}^{\mathrm{d}}(.)$ is the diagonal vector of the matrix (.).

All the matrices used in our formulation (i.e. P_k^+, P_k^-, L_k, Σ_k) are diagonals. Consequently, to save computation time, we must use a vectorial form for computing the various quantities of interest. The vectorial form of (12) and (13) are given by

$$\begin{aligned} &\hat{q}_k^+ = \hat{q}_k^- + \mathrm{vec}^{\mathrm{d}}(L_k) \circledast (\xi_k - \hat{q}_k^-) \\ &\mathrm{vec}^{\mathrm{d}}(L_k) = \mathrm{vec}^{\mathrm{d}}(P_k^-)//(\mathrm{vec}^{\mathrm{d}}(P_k^-) + \mathrm{vec}^{\mathrm{d}}(V_k)) \\ &\mathrm{vec}^{\mathrm{d}}(P_k^+) = \mathrm{vec}^{\mathrm{d}}(P_k^-) - \mathrm{vec}^{\mathrm{d}}(L_k) \circledast \mathrm{vec}^{\mathrm{d}}(P_k^-), \end{aligned} \tag{15}$$

where the symbol $\circledast$ stands for a componentwise product and // represents a componentwise divide. At each iteration k, HKA utilizes the relations (4), (6), (14), (15) and the next iteration is initialized with $m_k = q_k^+$, $\Sigma_k = \mathrm{diag}(S_k^+)^2$ and $k = k + 1$, where the notation diag(.) represents a diagonal matrix with the vector (.) on the diagonal. The algorithm used for the minimization of the objective function $J(q)$ is presented hereafter.

Heuristic Kalman Algorithm

1. Initialization. *Choose N, N_ξ and α. Set $k = 0$, $\hat{q}_k^- = m_0$, $P_k^- = \Sigma_0$, $m_k = \hat{q}_k^-$, $\Sigma_k = P_k^-$.*

2. Random generator $\mathcal{N}(m_k, \Sigma_k)$. *Generate a sequence of N vectors $\mathbf{q}(k) = \left\{q_k^1,\ q_k^2,\ \cdots,\ q_k^N\right\}$ according to a Gaussian distribution parametrized by m_k and Σ_k.*

3. Measurement process. *Using relations (4) and (6), compute* ξ_k *and* $\mathrm{vec^d}(V_k)$.

4. Optimal estimation. *Using relation (15), compute* $\mathrm{vec^d}(L_k)$, $\hat{q}_k^+$ *and* $\mathrm{vec^d}(P_k^+)$. *Using relation (14), compute* S_k^+.

5. Initialization of the next step. *Set* $\hat{q}_k^- = \hat{q}_k^+$, $P_k^- = \mathrm{diag}(S_k^+)^2$, $m_k = \hat{q}_k^-$, $\Sigma_k = P_k^-$, $k = k + 1$.

6. Termination test. *If the Stopping rule (see paragraph 2.2.2) is not satisfied, go to step 2, otherwise stop.*

The practical implementation of this algorithm requires

- properly initializing the Gaussian distribution, i.e. m_0 and Σ_0.
- selecting the user-defined parameters, namely N, N_ξ and α.
- introducing a stopping rule.

These various aspects are considered in sections 2.2.1 and 2.2.2.

2.2.1 Initialization and parameter settings

The initial parameters of the Gaussian generator are selected to cover the entire search space. To this end, the following rule can be used:

$$m_0 = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_{n_q} \end{bmatrix}, \; \Sigma_0 = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{n_q}^2 \end{bmatrix}, \quad \text{with:} \begin{cases} \mu_i = \dfrac{\bar{x}_i + \underline{x}_i}{2} \\ \sigma_i = \dfrac{\bar{x}_i - \underline{x}_i}{6} \end{cases}, \quad i = 1, \ldots, n_q, \quad (16)$$

where $\bar{x}_i$ (respectively, $\underline{x}_i$) is the i^{th} upper bound (respectively, lower bound) of the hyperbox search domain. With this rule, 99% of the samples are generated in the interval $\mu_i \pm 3\sigma_i$, $i = 1, \ldots, n_x$. The three following parameters must be set: the number of points

N, the number of best candidates N_ξ, and the coefficient α. To facilitate this task, TAB. 1 summarizes the influence of these parameters on the number of function evaluations (and thus on the CPU time) and on the average error.

Table 1: Effect of HKA parameters ($\nearrow$: increase, $\searrow$: decrease).

Parameter	$N \nearrow$	$N_\xi \nearrow$	$\alpha \nearrow$
Number of function evaluations	$\nearrow$	$\nearrow$	$\searrow$
Average error	$\searrow$	$\searrow$	$\nearrow$

2.2.2 Stopping rule

The algorithm stops when a given number of iterations, `MaxIter`, is reached (`MaxIter = 300` in all our experiments) or a given accuracy indicator is obtained. The latter accounts for the dispersion of the N_ξ best points. To this end, we consider that no significant improvement can be made when the N_ξ best points are in a ball of a given radius ρ ($\rho = 0.005$ in all our experiments). More precisely, the algorithm stops when $\max_{2 \leqslant i \leqslant N_\xi} ||q_1 - q_i||_2 \leqslant \rho$, where q_i $(i = 1, \ldots, N_\xi)$ are the N_ξ best candidates.

In conclusion, the search procedure HKA is articulated around three main components, the pdf function $f_k(q)$ (parametrized by m_k and Σ_k), the measurement process, and the Kalman estimator. Sampling from the pdf $f_k(q)$ at iteration k, creates a collection of vectors $\mathbf{q}(k)$. This collection is then used by the measurement process to provide information on the optimum. Via the Kalman estimator, this information is then combined with the pdf $f_k(q)$ in order to produce a new pdf $f_{k+1}(q)$ which will be used in the next iteration. Since the estimation process is carried out to minimize the variance of the posterior estimation error, it follows that Σ_k tends toward zero after a sufficient number of iterations. The resulting m_k can then be considered as a reliable estimate of the optimum. More precisely, an interesting property of this algorithm is its almost sure convergence to a near-optimal solution. This property is established in the next section.

2.3 Convergence properties

Let $J(q) : \mathbf{R}^{n_q} \to \mathbf{R}$ and $\mathcal{D} \subset \mathbf{R}^{n_q}$ be the cost function and the set of admissible solutions, respectively. A vector $q_{opt} \in \mathbf{R}^{n_q}$ is a global minimizer on $\mathcal{D}$, if:

$$q_{opt} \in \mathcal{D}, \quad \text{and} \quad J(q_{opt}) \leqslant J(q), \quad \forall q \in \mathcal{D}. \tag{17}$$

By definition, a set of vectors $q \in \mathcal{D}$ that belong to a neighborhood of the global minimum is a set of approximate solutions. Given $\varepsilon > 0$, the set of ε-approximate solutions of $J(q_{opt})$ is defined as:

$$\mathcal{E} = \{q \in \mathcal{D} : |J(q_{opt}) - J(q)| \leqslant \varepsilon\} \tag{18}$$

Thus any value $\hat{J}_{opt} = J(q)$ with $q \in \mathcal{E}$ is an estimate of the optimum $J(q_{opt})$ with a precision level ε, and every $q \in \mathcal{E}$ is a near-optimal solution.

In what follows, $f_k(q)$ represents the probability density function (pdf) of the random generator at iteration k. We denote by $\eta(k) = |\mathbf{q}(k) \cap \mathcal{E}|$ the cardinal number of the finite discrete set $\mathbf{q}(k) \cap \mathcal{E}$, i.e. the number of elements at iteration k. The proposition given hereafter gives the sufficient conditions for convergence of HKA to a near-optimal solution.

Proposition. *If the following two conditions are satisfied:*

1. $\mathcal{E}$ *is a convex set,*

2. $f_k(q) > 0$ *for all* $q \in \mathcal{D}$ *and* $k > 0$*,*

then the HKA converges almost surely to a near-optimal solution.

Proof. To show the almost sure convergence, it is sufficient to show that the probability of drawing at least N_ξ samples in $\mathcal{E}$ tends to unity as the number of iterations increases. For iteration k, the probability of drawing at least N_ξ samples in $\mathcal{E}$ is given by the binomial law

$$\mathbf{Pr}\{\eta(k) \geqslant N_\xi\} = z_k = \sum_{i=N_\xi}^{N} \frac{N!}{i!(N-i)!} p_k^i (1-p_k)^{N-i}, \tag{19}$$

where p_k is the probability that $x \in \mathcal{D}$ belongs to $\mathcal{E}$. This probability is given by

$$p_k = \int_{\mathcal{E}} f_k(q)dq. \tag{20}$$

Condition 2 implies that $p_k > 0$ for all k, and thus z_k is also a non-zero probability for all k. Therefore, a worst-case probability z_{wc} exists, which can be defined as follows:

$$0 < z_{wc} \leqslant z_k \quad \forall k. \tag{21}$$

Since $\mathcal{E}$ is convex, $\mathbf{Pr}\{\eta(k) \geqslant N_\xi\}$ is also the probability that ξ_k belongs to $\mathcal{E}$. Consider the sequence $\{\xi_k^{best}\}$ defined as

$$\begin{aligned} \xi_{k+1}^{best} &= \xi_{k+1} \text{ if } J(\xi_{k+1}) < J(\xi_k) \\ \xi_{k+1}^{best} &= \xi_k^{best} \text{ if } J(\xi_{k+1}) \geqslant J(\xi_k). \end{aligned} \tag{22}$$

The almost sure convergence means that $\lim_{k\to\infty} \mathbf{Pr}\left\{\xi_k^{best} \in \mathcal{E}\right\} = 1$. Then, we must show that the sequence $\{\xi_k^{best}\}$ obtained by HKA converges with a probability of one to $\mathcal{E}$. To this end, consider the random variable y_i, $i = 2, \cdots, k$, defined as

$$\begin{aligned} y_i &= 1, \text{ if } J(\xi_i^{best}) \leqslant J(\xi_{i-1}^{best}) - \varepsilon \\ y_i &= 0, \text{ if } J(\xi_i^{best}) > J(\xi_{i-1}^{best}) - \varepsilon \end{aligned} \tag{23}$$

In other words, $y_i = 1$ means a success in our attempt to reduce the cost function J by at least ε. Using the value of J for the initial measurement (i.e. ξ_1), we introduce the following variable:

$$N_s = \left\lfloor \frac{J(\xi_1) - J(q_{opt})}{\varepsilon} \right\rfloor, \tag{24}$$

where $\lfloor v \rfloor$ is the largest integer smaller than v. Therefore, a sufficient condition so that ξ_k^{best} belongs to $\mathcal{E}$ is

$$\sum_{i=2}^{k} y_i \geqslant N_s + 1. \tag{25}$$

The probability $\mathbf{Pr}\{\xi_k^{best} \notin \mathcal{E}\}$ is less than or equal to the probability that the number of successful steps does not exceed N_s:

$$\mathbf{Pr}\{\xi_k^{best} \notin \mathcal{E}\} \leqslant \mathbf{Pr}\left\{\sum_{i=2}^{k} y_i \leqslant N_s\right\}. \tag{26}$$

The latter probability increases with a decrease in the probability of successful steps. From (21), we know that the probability of any successful step is greater than or equal to the worst case probability z_{wc}. Consequently, we can write:

$$\mathbf{Pr}\left\{\sum_{i=2}^{k} y_i \leqslant N_s\right\} \leqslant \sum_{i=0}^{N_s} \frac{k!}{i!(k-i)!} z_{wc}^i (1-z_{wc})^{k-i}, \tag{27}$$

this latter expression can be upper-bounded as follows (see [20]):

$$\sum_{i=0}^{N_s} \frac{k!}{i!(k-i)!} z_{wc}^i (1-z_{wc})^{k-i} \leqslant \frac{N_s+1}{N_s!} k^{N_s} (1-z_{wc})^k. \tag{28}$$

Therefore, from (26), we have

$$\mathbf{Pr}\{\xi_k^{best} \notin \mathcal{E}\} \leqslant \frac{N_s+1}{N_s!} k^{N_s} (1-z_{wc})^k. \tag{29}$$

Since $z_{wc} > 0$, it is clear that

$$\lim_{k\to\infty} k^{N_s} (1-z_{wc})^k = 0, \tag{30}$$

thus $\lim_{k\to\infty} \mathbf{Pr}\{\xi_k^{best} \in \mathcal{E}\} = 1$. This shows the almost sure convergence of HKA.

3 Numerical experiments

In this section, the ability of the presented method to solve a wide range of non-convex optimization problems is tested on various numerical examples, both in the unconstrained and constrained cases. In all cases, our results were compared to those obtained via other metaheuristics. We did not program the corresponding algorithms, but only used the available published results. Details on these metaheuristics are given in the cited literature. We considered the unconstrained and constrained cases separately, because specific methods have been proposed to handle constraints (in particular the notion of co-evolution or the introduction of an augmented Lagrangian). The same HKA algorithm was used in both cases. The constraints were handled merely by introducing an augmented cost function using penalty functions. The various experiments were performed using a 1.2 Ghz Celeron personal computer.

3.1 Unconstrained case

HKA was compared to other metaheuristics such as $ACO_{\mathbf{R}}$, CGA, ECTS, ESA and INTEROPT, which are listed in TAB. 2. The efficiency of HKA was tested using a set of well-known test functions (RC, B2, DJ, $S_{4,5}$, $S_{4,7}$, $S_{4,10}$, and $H_{6,4}$), which are listed in the Appendix. For these experiments, we performed each test 100 times and we compared

Table 2: List of the methods used in our comparisons.

Method	Reference
Ant colony optimization for continuous domains ($ACO_{\mathbf{R}}$)	K. Socha and M. Dorigo [26]
Continuous Genetic Algorithm (CGA)	R. Chelouah and P. Siarry [4]
Enhanced Continuous Tabu Search (ECTS)	R. Chelouah and P. Siarry [3]
Enhanced Simulated Annealing (ESA)	P. Siarry and al. [25]
INTEROPT	G. L. Bilbro and W.E. Snyder [1]

our results with those that have been previously published. In all these experiments, the following parameters were used: number of points $N = 25$, number of best candidates $N_\xi = 5$, slowdown coefficient $\alpha = 0.9$. The experimental results are presented in TAB. 3. For each test function, we give the success ratio for 100 runs and the corresponding average number of function evaluations. It can be seen that some results are not available for ECTS, ESA and INTEROPT (as indicated by the symbol " - "). As shown in TAB. 3, the best results were obtained for $ACO_{\mathbf{R}}$, CGA, and HKA. The number of evaluations produced by HKA was slightly greater than those produced by CGA and $ACO_{\mathbf{R}}$, but its success ratio was better. TAB. 4 presents the results on average error. These results were

Table 3: Comparison of HKA with $ACO_{\mathbf{R}}$, CGA, ECTS, ESA and INTEROPT.

	Success Ratio (%)						Average number of function evaluations					
	ACO_R	CGA	ECTS	ESA	INTER OPT	HKA	ACO_R	CGA	ECTS	ESA	INTER OPT	HKA
RC	100	100	100	-	100	100	857	620	245	-	4172	625
B2	100	100	-	-	-	100	559	430	-	-	-	1275
DJ	100	100	-	-	-	100	392	750	-	-	-	600
$S_{4,5}$	57	76	75	54	40	93	793	610	825	1137	3700	675
$S_{4,7}$	79	83	80	54	60	92	748	680	910	1223	2426	686
$S_{4,10}$	81	81	75	50	50	93	715	650	898	1189	3463	687
$H_{6,4}$	100	100	100	100	100	97	722	976	1520	2638	17262	667
Mean Values	89	92	86	65	70	97	684	674	880	1547	6205	745

not available for ACOR and INTEROPT. Compared to CGA, ECTS and ESA, HKA led to a lower average error.

Table 4: Average error.

	Average Error					
	ACO_R	**CGA**	**ECTS**	**ESA**	**INTER OPT**	**HKA**
RC	-	1.0e-4	5.0e-2	-	-	5.0e-6
B2	-	3.0e-4	-	-	-	4.0e-5
DJ	-	2.0e-4	3.0e-8	-	-	2.0e-6
$S_{4,5}$	-	1.4 e-1	1.0e-2	4.2e-3	-	2.0e-4
$S_{4,7}$	-	1.2 e-1	1.0e-2	8.4e-3	-	8.0e-4
$S_{4,10}$	-	1.5 e-1	1.0e-2	4.3e-2	-	5.0e-4
$H_{6,4}$	-	4.0e-2	5.0e-2	5.9e-2	-	8.0e-3
Mean Values	-	6.4e-2	2.2e-2	2.9e-2	-	1.4e-3

3.2 Constrained case

HKA was compared to other metaheuristics specifically designed for solving constrained problems. Two examples were considered, the welded beam design problem and the robust PID design problem. In both experiments, HKA handled constraints via a new objective function which includes penalty functions:

$$J_{new}(x) = J(x) + w \sum_{i=1}^{N_c} \max(g_i(x), 0), \tag{31}$$

where N_C is the number of constraints, $g_i(x)$ is the i^{th} inequality constraint of the form $g_i(x) \leqslant 0$, and w is a weighting factor, which was set to 100 in both experiments.

The welded beam design problem. A welded beam is designed for minimum cost subject to constraints on shear stress $\tau(x)$, bending stress in the beam $\sigma(x)$, buckling load on the bar Pc, end deflection of the beam $\delta(x)$, and side constraints [15]. There are four design variables as shown in FIG. 3: $h\,(x_1)$, $l\,(x_2)$, $t\,(x_3)$ and $b\,(x_4)$, $x = [x_1 \quad x_2 \quad x_3 \quad x_4]^T$.

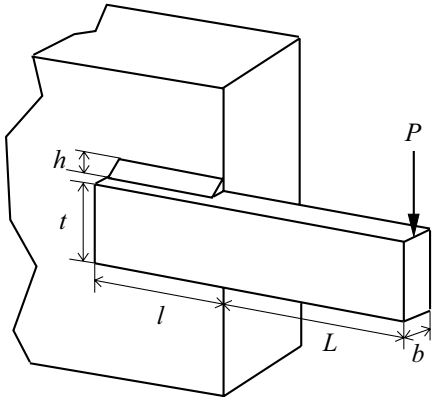

Figure 3: Welded beam design problem.

The problem can be mathematically formulated as follows:

$$\begin{aligned}
&\text{Minimize} && J(x) = (1+c_1)x_1^2 x_2 + c_2 x_3 x_4 (L + x_2) \\
&\text{Subject to:} && g_1(x) = \tau(x) - \tau_{max} \leqslant 0 \\
&&& g_2(x) = \sigma(x) - \sigma_{max} \leqslant 0 \\
&&& g_3(x) = x_1 - x_4 \leqslant 0 \\
&&& g_4(x) = c_1 x_1 + c_2 x_3 x_4 (L + x_2) - 5 \leqslant 0 \\
&&& g_5(x) = h_{min} - x_1 \leqslant 0 \\
&&& g_6(x) = \delta(x) - \delta_{max} \leqslant 0 \\
&&& g_7(x) = P - P_c(x) \leqslant 0
\end{aligned} \tag{32}$$

where

$$\begin{aligned}
&\tau(x) = \sqrt{\tau_1^2 + 2\tau_1\tau_2 \frac{x_2}{2R} + \tau_2^2}, \quad \tau_1 = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau_2 = \frac{MR}{I} \\
&M = P\left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad I = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\} \\
&\sigma(x) = \frac{6PL}{x_4 x_3^2}, \quad \delta(x) = \frac{4PL^3}{E x_3^2 x_4}, \quad P_c(x) = \frac{4.013E\sqrt{x_3^2 x_4^6/36}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)
\end{aligned} \tag{33}$$

and

$$c_1 = 0.10471, \quad c_2 = 0.04811, \quad P = 6 \times 10^3, \quad L = 14, \quad E = 3 \times 10^7$$
$$G = 1.2 \times 10^7, \quad h_{min} = 0.125, \quad \delta_{max} = 0.25, \quad \tau_{max} = 1.36 \times 10^4, \quad \sigma_{max} = 3 \times 10^4. \tag{34}$$

The ranges of design variables are

$$0.1 \leqslant x_1 \leqslant 2, \quad 0.1 \leqslant x_2 \leqslant 10, \quad 0.1 \leqslant x_3 \leqslant 10, \quad 0.1 \leqslant x_4 \leqslant 2.$$

This problem has been solved using a genetic algorithm (GA) with binary representation and a traditional penalty function [8]. It has also been solved using geometric programming (GP) [23]. Recently, this problem was also solved using a GA-based co-evolution model [5] as well as a multi-objective genetic algorithm (MGA) [6, 7]. Even more recently, this problem was solved using a co-evolutionary particle swarm optimization (CPSO), with a better solution than those previously obtained [15].

In this experiment, we performed the minimization problem 30 times and we compared our results with those obtained via the methods listed in TAB. 5. The following parameters were used: number of points $N = 50$, number of best candidates $N_\xi = 5$, slowdown coefficient $\alpha = 0.3$.

Table 5: List of the methods used in our comparisons.

Method	Reference
Geometric Programming (GP)	K.M. Ragsdell and D.T. Phillips [23]
Genetic Algorithm and Penalty function (GAP)	K. Deb [8]
Co-Evolutionnary Genetic Algorithm (CEGA)	C.A.C. Coello [5, 6]
Multi-objective Genetic Algorithm (MGA)	C.A.C. Coello and E.M. Montes [7]
Co-evolutionary Particle Swarm Optimization (CPSO)	Q. He and L.Wang [15]

The best solutions obtained by the above-mentioned approaches are listed in TAB. 6 and the statistical results are shown in TAB. 7.

Table 6: Comparison of the best solution found using different methods.

	GP	GAP	CEGA	MGA	CPSO	HKA
x_1 (h)	0.245500	0.248900	0.208800	0.205986	0.202369	0.205624
x_2 (l)	6.196000	6.173000	3.420500	3.471328	3.544214	3.473825
x_3 (t)	8.273000	8.178900	8.997500	9.020224	9.048210	9.038561
x_4 (b)	0.245500	0.253300	0.210000	0.206480	0.205723	0.205738
$g_1(x)$	-5743.826517	-5758.603777	-0.337812	-0.074092	-12.839796	-5.621131
$g_2(x)$	-4.715097	-255.576901	-353.902604	-0.266227	-1.247467	-14.103308
$g_3(x)$	0.000000	-0.004400	-0.001200	-0.000495	-0.001498	-0.000115
$g_4(x)$	-3.020289	-2.982866	-3.411865	-3.430043	-3.429347	-3.432290
$g_5(x)$	-0.120500	-0.123900	-0.083800	-0.080986	-0.079381	-0.080624
$g_6(x)$	-0.234208	-0.234160	-0.235649	-0.235514	-0.235536	-0.235550
$g_7(x)$	-3604.275002	-4465.270928	-363.232384	-58.666440	-11.681355	-1.595159
$J(x)$	2.385937	2.433116	1.748309	1.728226	1.728024	1.7255393

Table 7: Statistical results.

Method	Best	Mean	Worst	Std Dev	Average number of function evaluations
GP	2.385937	-	-	-	-
GAP	2.433116	-	-	-	-
CEGA	1.748309	1.771973	1.785835	0.011220	900000
MGA	1.728226	1.792654	1.993408	0.074713	80000
CPSO	1.728024	1.748831	1.782143	0.012926	200000
HKA	1.725539	1.725824	1.726287	0.000172	18600

As shown in Tab. 6, the best feasible solution found by HKA was better than the best solutions found by other techniques. As shown in Tab. 7, the average searching quality of HKA was also significantly better than those of other methods, and even the worst solution found by HKA was better than the best solution found via CPSO method. In addition the standard deviations of the results obtained by HKA were very small. Furthermore, the number of function evaluations was significantly lower than those obtained by the other methods.

Robust PID controller tuning. In a wide range of engineering applications, the problem of designing a robust Proportional-Integral-Derivative (PID) controller remains an open issue. This is mainly due to the fact that the underlying optimization problem is non-convex and thus suffers from computational intractability and conservatism. To overcome

these difficulties, Kim et al. [17] suggested solving this problem by augmented Lagrangian Particle Swarm Optimization (ALPSO). Here we show that HKA is able to solve the problem of designing a robust PID controller. The obtained results were then compared to those of ALPSO. The problem can be mathematically formulated as follows

$$\begin{aligned}
\text{Minimize} \quad & J(x) = \arg\max_{\lambda_i(x)}\{\mathrm{Re}(\lambda_i(x)),\ \forall i\}, \quad x = [x_1 \quad x_2 \quad x_3 \quad x_4]^T \\
\text{Subject to:} \quad & g_1(x) = \sup_{\omega \geqslant 0} |[W_S(s)]_{s=j\omega}[S(s,x)]_{s=j\omega}]| - 1 \leqslant 0 \\
& g_2(x) = \sup_{\omega \geqslant 0} |[W_T(s)]_{s=j\omega}[T(s,x)]_{s=j\omega}]| - 1 \leqslant 0 \\
& \underline{x}_i \leqslant x_i \leqslant \bar{x}_i, \quad i = 1, \cdots, 4
\end{aligned} \tag{35}$$

where $x = [x_1 \quad x_2 \quad x_3 \quad x_4]^T$ is the vector of decision variables, $\underline{x}_i$ and $\bar{x}_i$ are the bounds of the hyperbox search domain, s is the Laplace variable, ω is the frequency (rad/s), j is the unit imaginary number, $S(s,x)$ is the sensitivity function defined as $S(s,x) = 1/(1+L(s,x))$, $T(s,x)$ is the closed-loop system defined as $T(s,x) = L(s,x)/(1+L(s,x))$, $L(s,x)$ is the open-loop transfer function defined as $L(s,x) = P(s)K(s,x)$ where $P(s)$ is the transfer function of the system to be controlled, and $K(s,x)$ the transfer function of the PID controller

$$K(s,x) = 10^{x_1}\left(1 + \frac{1}{10^{x_2}s} + \frac{10^{x_3}s}{1+10^{(x_3-x_4)}s}\right) \tag{36}$$

In the objective function, $\lambda_i(x)$ denotes the i^{th} pole of the closed-loop system. The frequency-dependent weighting functions $W_S(s)$ and $W_T(s)$ are set so as to meet the performance specifications of the closed-loop system.

As in [17], we solved this optimization problem for the magnetic levitation system described in [27]. The process model is defined as

$$P(s) = \frac{7.147}{(s-22.55)(s+20.9)(s+13.99)}. \tag{37}$$

The frequency-dependent weighting functions $W_S(s)$ and $W_T(s)$ are respectively given as

$$W_S(s) = \frac{5}{s+0.1}, \quad W_T(s) = \frac{43.867(s+0.066)(s+31.4)(s+88)}{(s+10^4)^2}. \tag{38}$$

The search space is

$$2 \leqslant x_1 \leqslant 4, \quad -1 \leqslant x_2 \leqslant 1, \quad -1 \leqslant x_3 \leqslant 1, \quad 1 \leqslant x_4 \leqslant 3.$$

In this test, we performed minimization 30 times and we compared our results with those presented in [17]. The following parameters were used: number of points $N = 50$, number of best candidates $N_\xi = 5$, slowdown coefficient $\alpha = 0.4$.

The best solutions obtained via ALPSO and HKA are listed in TAB. 8 and the statistical results are shown in TAB. 9 (the statistical results were not available for ALPSO).

Table 8: Comparison of the best solutions found via ALPSO and HKA.

	ALPSO	**HKA**
x_1	3.2548	3.2542
x_2	-0.8424	-0.8634
x_3	-0.7501	-0.7493
x_4	2.3137	2.3139
$g_1(x)$	6.1e-3	-9.6e-4
$g_2(x)$	-4.0e-4	-1.2e-3
$J(x)$	-1.7197	-1.7106

Table 9: Statistical results.

Method	**Best**	**Mean**	**Worst**	**Std Dev**	**CPU time**	**Average number of function evaluations**
ALPSO	-1.7197	-	-	-	687 s	25000
HKA	-1.7106	-1.7023	-1.6891	0.0048	266 s	5427

The higher absolute value of the objective function obtained using ALPSO is due to the violation of constraint $g_1(x)$; this is not the case for our solution in which all constraints are satisfied. As shown in TAB. 9, HKA required a smaller number of function evaluations and had a lower associated CPU time compared to ALPSO.

If, as in ALPSO, a small violation of the constraint $g_1(x)$ is tolerated, we obtained the results listed in TAB. 10 and TAB. 11.

Table 10: Comparison of the best solutions found via ALPSO and HKA.

	ALPSO	HKA
x_1	3.2548	3.2556
x_2	-0.8424	-0.8354
x_3	-0.7501	-0.7539
x_4	2.3137	2.3127
$g_1(x)$	6.1e-3	4.9e-3
$g_2(x)$	-4.0e-4	-2.8e-3
$J(x)$	-1.7197	-1.7435

Table 11: Statistical results.

Method	Best	Mean	Worst	Std Dev	CPU time	Average number of function evaluations
ALPSO	-1.7197	-	-	-	687 s	25000
HKA	-1.7435	-1.7381	-1.7323	0.0030	248 s	5072

The best solution found by HKA was better than the solution found by ALPSO with, in addition, a smaller violation constraint (TAB. 10). TAB. 11 shows that the worst solution found by HKA was better than the solution found via ALPSO; in addition, HKA had a smaller number of function evaluations (and thus a lower corresponding CPU time) compared to ALPSO.

4 Conclusion

In this paper, a new optimization algorithm, called the heuristic Kalman algorithm (HKA), was presented. The main characteristic of HKA is to explore the search space via a Gaussian pdf. This exploration is directed by appropriately adjusting the pdf parameters so that they converge to a near-optimal solution with low variance. To this end, a measurement process followed by a Kalman estimator is introduced. The role of the Kalman estimator is to combine the prior pdf function with the measurement process to give a new pdf function

for the exploration of the search space. We demonstrated that under not too restrictive conditions, the HKA algorithm converges "almost surely" to a near-optimal solution.

We tested the performance of HKA using several non-convex test problems, both in the unconstrained and constrained cases. The results obtained show that HKA can be considered as a good alternative for solving difficult non-convex problems quickly and with a high probability of success.

Appendix

Test functions RC, B2, DJ, $S_{4,5}$, $S_{4,7}$, $S_{4,10}$, and $H_{6,4}$

- **Branin's function** (RC) (2 variables)

$$J(x) = \left(x_2 - \left(\frac{5.1}{4\pi^2}\right)x_1^2 + \left(\frac{5}{\pi}\right)x_1 - 6\right)^2 + 10\left(1 - \left(\frac{1}{8\pi}\right)\right)\cos(x_1) + 10$$

  search domain: $-5 \leqslant x_i \leqslant 10$, $i = 1, 2$; 3 global minima: $x_{opt} = (-\pi, 12.275)$, $(\pi, 2.275)$, $(9.42478, 2.475)$, $J(x_{opt}) = 0.397887$.

- **Bohachecsky's function** (B2) (2 variables)

$$J(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7;$$

  search domain: $-100 \leqslant x_i \leqslant 100$, $i = 1, 2$; global minimum: $x_{opt} = (0, 0)$, $J(x_{opt}) = 0$.

- **De Jong's function** (DJ) (3 variables);$J(x) = x_1^2 + x_2^2 + x_3^2$; search domain: $-5 \leqslant x_i \leqslant 5$, $i = 1, 3$; global minimum: $x_{opt} = (0, 0, 0)$, $J(x_{opt}) = 0$.

- **Shekel's functions** ($S_{4,5}$, $S_{4,7}$, $S_{4,10}$) (4 variables); search domain: $0 \leqslant x_i \leqslant 9$, $i = 1, 4$; 3 functions were considered $S_{4,5}$, $S_{4,7}$ and $S_{4,10}$. For a complete definition of these functions see [26].

- **Hartmann's functions** ($H_{6,4}$) (6 variables); search domain: $0 \leqslant x_i \leqslant 1$, $i = 1, 6$; global minimum: $J(x_{opt}) = -3.322368$. For a complete definition of this function, see [26].

References

[1] G. L. Bilbro and W.E. Snyder. Optimization of functions with many minima. *IEEE Transactions on Systems, Man, and Cybernetics*, 24:840–849, 1991.

[2] S. Boyd and L. Vandenberghen. *Convex Optimization.* Cambridge University Press, 2004.

[3] R. Chelouah and P. Siarry. *Enhanced Continuous Tabu Search: An Algorithm for the Global Optimization of Multiminima Function.* In S. Voss, S. Martello, I.H. Osman and C. Roucairol (eds.), Meta-Heuristics, Advances and Trends in Local Search Paradigms for Optimization. Kluwer Academic Publishers. Chap. 4, pp. 49-61, 1999.

[4] R. Chelouah and P. Siarry. A continuous genetic algorithm designed for the global optimization of multimodal functions. *Journal of Heuristics*, 6:191–213, 2000.

[5] C.A.C Coello. Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, 41:113–127, 2000.

[6] C.A.C Coello. Theoretical and numerical constraint handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191:1245–1287, 2002.

[7] C.A.C Coello and E.M. Montes. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16:193–203, 2002.

[8] K. Deb. Optimal design of a welded beam via genetic algorithms. *AIAA Journal*, 29:2013–2015, 1991.

[9] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29–41, 1996.

[10] R.C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan*, pages 39–43, 1995.

[11] D.B. Fogel. *Evolutionary computation: towards a new philosophy of machine intelligence, 3rd edition.* Wiley-IEEE Press, 2006.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA, 1979.

[13] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Kluwer Academic Publishers, Boston, MA, 1989.

[14] F. G. Guimarães, R. M. Palhares, F. Campelo, and H. Igarashi. Design of mixed image control systems using algorithms inspired by the immune system. *Information Sciences*, 177:4368–4386, 2007.

[15] Q. He and L. Wang. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artifical Intelligence*, 20:89–99, 2007.

[16] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[17] T.H. Kim, I. Maruta, and T. Sugie. Robust pid controller tuning based on the constrained particle swarm optimization. *Automatica*, 44:1104–1110, 2008.

[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[19] Y. Liu, Z. Yi, H. Wu, M. Ye, and K. Chen. A tabu search approach for the minimum sum-of-squares clustering problem. *Information Sciences*, 178(12):2680–2704, 2008.

[20] J. Matyas. Random optimization. *Automation and Remote control, translated from Avtomatika i Telemekhanika*, 26(2):246–253, 1965.

[21] P. S. Maybeck. *Stochastic models, estimation, and control*. Academic Press, 1979.

[22] Z. Michalewicz. Genetic algorithms, numerical optimization and constraints. *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 151–158, 1995.

[23] K.M. Ragsdell and D.T. Phillips. Optimal design of a class of welded structures using geometric programming. *ASME Journal of Engineering for Industries*, 98:1021–1025, 1976.

[24] R. T. Rockafellar. Lagrange multipliers and optimality. *SIAM Review*, 35:183–238, 1993.

[25] P. Siarry, G. Berthiau, F. Durbin, and J. Haussy. Enhanced simulated annealing for globally minimizing functions of many continuous variables. *ACM Transactions on Mathematical Software*, 23:209–228, 1997.

[26] K. Socha and M. Dorigo. Ant colony optimization for continuous domains. *European Journal of Operational Research*, 185:1155–1173, 2008.

[27] T. Sugi, K. Simizu, and J. Imura. Hinf control with exact linearization and its applications to magnetic levitation systems. *In IFAC 12th World congress*, 4:363–366, 1993.

[28] F. van den Bergh and A.P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176:937–971, 2006.

[29] X. Wang, X. Z. Gao, and S. J. Ovaska. Artificial immune optimization methods and applications - a survey. *Proceedings of the IEEE International Conference On Systems, Man and Cybernetics*, 4:3415–3420, 2004.