STRUCTURED CONTROLLER DESIGN WITH EVOLUTIONARY OPTIMIZATION

403-26

Tony Wong, Pascal Bigras, Pascal Côté
Department of automated manufacturing engineering
École de technologie supérieure – University of Québec
1100 Notre Dame West, Montréal (Québec)
Canada, H3C 1K3

Abstract

A set of standard evolution strategies is implemented in order to find the optimal tuning of a structured controller. The structured controller design problem is a non-convex optimization problem. The usual solution technique involves the use of branch and bound global search algorithm. Standard evolution strategies are able to find the global optimum with comparable solution quality, less overhead and considerable gain in computation time.

Key Words
evolutionary strategies, structured controller design, non-convex optimization, H_2 minimization, PID controller.

1. Introduction

The goal of this paper is to provide fast and accurate solutions, using the evolutionary approach, to a structured controller design problem. A structured controller is a class of controllers that are neither state feedback controllers nor full–order output feedback controllers [1]. This class of controllers is suitable for industrial applications since its order and structure can be imposed arbitrary. The controller design problem is first formulated as a non-convex optimization problem that minimizes a matching model criterion. Then, it is solved by a set of standard evolution strategies (ESs) without any specialized extension.

Over the past decades, there are many advances in the field of linear optimal control. In particular, several methods based on the Linear Matrix Inequality (LMI) formulation have been proposed for optimal control problems with state feedback and full-order output feedback controllers [2]. More recently, optimal static output feedback, fixed order controllers and structured controllers have became the focus of great research effort. The design problem is a non-convex optimization problem. The optimal static output feedback and structured controller control problem can be solved by using local solution based on multiple LMI problem solutions [3, 4]. However, branch and bound (BB) algorithm is still needed to find the global minima [5]. As shown in this work, the application of evolution strategy (ES) can yield comparable results with considerable gain in computation time.

The outline of this paper is as follows: Section 2 introduces the system model used in this work. Section 3 presents the optimization problem formulation. Section 4 covers the standard ES optimizers used to solve the non-convex optimization. In Section 5, the evolutionary approach is applied to optimally tune the PID controller gains for arbitrary plant, sensors and reference model. Finally, conclusions are drawn in section 6.

2. System Model

Continuous linear time-invariant feedback control systems are generally characterized by a plant, sensors, and a feedback controller. In this paper, it is assumed that the controller is structured and has a fixed order. The gain adjustment that minimizes the difference between a reference model and the closed loop of the feedback control system results in a non-convex optimization problem.

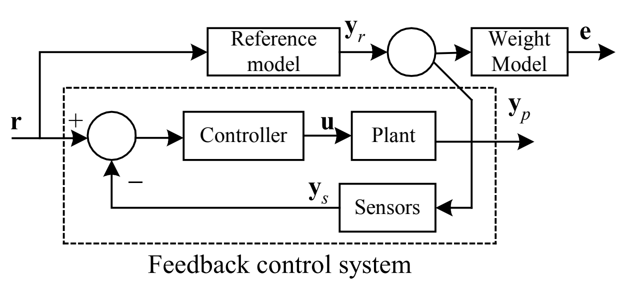

Figure 1 Blocks-diagram of the system

The blocks-diagram corresponding to this system is illustrated by Fig. 1. Note that there exists a weight error model between the reference model and the closed loop system. This provides a more flexible control system. In Fig. 1 the reference model is described in state-space form by

$$\begin{aligned} \dot{\mathbf{x}}_r &= \mathbf{A}_r \mathbf{x}_r + \mathbf{B}_r \mathbf{r}, \\ \mathbf{y}_r &= \mathbf{C}_r \mathbf{x}_r, \end{aligned} \tag{1}$$

where $\mathbf{x}_r \in \mathcal{R}^{n_r}$, $\mathbf{y}_r \in \mathcal{R}^{n_y}$ and $\mathbf{r} \in \mathcal{R}^{n_y}$. The plant model is itself described by

$$\begin{aligned}\dot{\mathbf{x}}_p &= \mathbf{A}_p\mathbf{x}_p + \mathbf{B}_p\mathbf{u},\\ \mathbf{y}_p &= \mathbf{C}_p\mathbf{x}_p,\end{aligned} \tag{2}$$

where $\mathbf{x}_p \in \mathcal{R}^{n_p}$, $\mathbf{y}_p \in \mathcal{R}^{n_y}$ and $\mathbf{u} \in \mathcal{R}^{n_u}$. Similarly the sensors model is described as

$$\begin{aligned}\dot{\mathbf{x}}_s &= \mathbf{A}_s\mathbf{x}_s + \mathbf{B}_s\mathbf{y}_p,\\ \mathbf{y}_s &= \mathbf{C}_s\mathbf{x}_s + \mathbf{D}_s\mathbf{y}_p,\end{aligned} \tag{3}$$

where $\mathbf{x}_s \in \mathcal{R}^{n_s}$, $\mathbf{y}_s \in \mathcal{R}^{n_y}$. The weight error model is given by

$$\begin{aligned}\dot{\mathbf{x}}_w &= \mathbf{A}_w\mathbf{x}_w + \mathbf{B}_w\varepsilon,\\ \mathbf{e} &= \mathbf{C}_w\mathbf{x}_w,\end{aligned} \tag{4}$$

where $\varepsilon = \mathbf{y}_r - \mathbf{y}_p = \mathbf{C}_r\mathbf{x}_r - \mathbf{C}_p\mathbf{x}_p$, $\mathbf{x}_w \in \mathcal{R}^{n_w}$ and $\mathbf{e} \in \mathcal{R}^{n_y}$. Finally the structured controller model is described by

$$\begin{aligned}\dot{\mathbf{x}}_c &= \mathbf{A}_c(\mathbf{k})\mathbf{x}_c + \mathbf{B}_c(\mathbf{k})(\mathbf{r}-\mathbf{y}_s),\\ \mathbf{u} &= \mathbf{C}_c(\mathbf{k})\mathbf{x}_c + \mathbf{D}_c(\mathbf{k})(\mathbf{r}-\mathbf{y}_s),\end{aligned} \tag{5}$$

where $\mathbf{x}_c \in \mathcal{R}^{n_c}$ and according to [1] the structured controller is rearranged as

$$\begin{aligned}\begin{bmatrix}\mathbf{A}_c(\mathbf{k}) & \mathbf{B}_c(\mathbf{k})\\ \mathbf{C}_c(\mathbf{k}) & \mathbf{D}_c(\mathbf{k})\end{bmatrix} &= \begin{bmatrix}\mathbf{A}_{c0} & \mathbf{B}_{c0}\\ \mathbf{C}_{c0} & \mathbf{D}_{c0}\end{bmatrix} +\\ \mathbf{\Theta}_L\mathbf{K}_s(\mathbf{k})\mathbf{\Theta}_R &= \begin{bmatrix}\mathbf{A}_{c0} & \mathbf{B}_{c0}\\ \mathbf{C}_{c0} & \mathbf{D}_{c0}\end{bmatrix} + \sum_{i=1}^{n_k}\Theta_{L_i}k_i\Theta_{R_i},\end{aligned} \tag{6}$$

where $\mathbf{k} \in \mathcal{R}^{n_k}$ is the gain vector, $\Theta_{L_i} \in \mathcal{R}^{n_u \times n_{k_i}}$ and $\Theta_{R_i} \in \mathcal{R}^{n_{k_i} \times n_s}$ are full rank matrices. The system described by Eqs (1), (2), (4), (5) and (6) can be rewritten in one state-space model as

$$\begin{aligned}\dot{\mathbf{x}} &= (\mathcal{A} + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{C}_u)\mathbf{x} + (\mathcal{B}_r + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{D}_u)\mathbf{r},\\ \mathbf{e} &= \mathcal{C}\mathbf{x},\end{aligned} \tag{7}$$

where

$$\mathcal{A} = \begin{bmatrix}\mathbf{A}_r & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{A}_{c0} & -\mathbf{B}_{c0}\mathbf{C}_s & \mathbf{0} & -\mathbf{B}_{c0}\mathbf{D}_s\mathbf{C}_p\\ \mathbf{0} & \mathbf{0} & \mathbf{A}_s & \mathbf{0} & \mathbf{B}_s\mathbf{C}_p\\ \mathbf{B}_w\mathbf{C}_r & \mathbf{0} & \mathbf{0} & \mathbf{A}_w & -\mathbf{B}_w\mathbf{C}_p\\ \mathbf{0} & \mathbf{B}_p\mathbf{C}_{c0} & -\mathbf{B}_p\mathbf{D}_{c0}\mathbf{C}_s & \mathbf{0} & \mathbf{A}_p - \mathbf{B}_p\mathbf{D}_{c0}\mathbf{D}_s\mathbf{C}_p\end{bmatrix},$$

and

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_r\\ \mathbf{x}_c\\ \mathbf{x}_m\\ \mathbf{x}_w\\ \mathbf{x}_p\end{bmatrix}, \mathcal{B}_u = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{B}_p\end{bmatrix}, \mathcal{B}_r = \begin{bmatrix}\mathbf{B}_r\\ \mathbf{B}_{c0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{B}_p\mathbf{D}_{c0}\end{bmatrix}, \mathcal{D}_u = \begin{bmatrix}\mathbf{0}\\ \mathbf{I}\end{bmatrix},$$

$$\mathcal{C}_u^T = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{I} & \mathbf{0}\\ \mathbf{0} & -\mathbf{C}_s^T\mathbf{D}_c^T\\ \mathbf{0} & \mathbf{0}\\ \mathbf{0} & -\mathbf{C}_p^T\mathbf{D}_s^T\end{bmatrix}, \mathcal{C}^T = \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{C}_w^T\\ \mathbf{0}\end{bmatrix}.$$

3. Optimization problem

According to the closed loop system described by (7), the goal is to find the controller gain vector **k** that minimizes the error **e**. This optimization problem can be formulated as

$$\mathbf{k} = \arg\min \|\mathbf{H}_{\varepsilon r}\|_2 \tag{8}$$

where

$$\mathbf{H}_{\varepsilon r}(s) = \mathcal{C}(s\mathbf{I} - (\mathcal{A} + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{C}_u))^{-1}(\mathcal{B}_r + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{D}_u)$$

and $\|\mathbf{H}\|_2$ is the 2-norm of system **H** that exists only if **H** is Hurwitz-stable. According to [1], $\|\mathbf{H}\|_2$ is defined by

$$\|\mathbf{H}\|_2^2 = \frac{1}{2\pi}\,\mathrm{trace}\int_{-\infty}^{\infty}\mathbf{H}(j\omega)\mathbf{H}(j\omega)^*\,d\omega\,.$$

In order to avoid the integral evaluation, the minimization problem given by (8) can be rewritten as

$$\mathbf{k} = \arg\min(\mathrm{trace}(\mathcal{L}_1(\mathcal{B}_r + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{D}_u, \mathbf{P}))) \tag{9}$$

where **P**, a symmetric semi-positive definite matrix, is the solution of the following equation:

$$\mathcal{L}_2((\mathcal{A} + \mathcal{B}_u\mathbf{K}_s(\mathbf{k})\mathcal{C}_u), \mathbf{P}) + \mathbf{C}^T\mathbf{C} = \mathbf{0}, \tag{10}$$

where

$$\mathcal{L}_1(\mathbf{B}, \mathbf{P}) = \mathbf{B}^T\mathbf{P}\mathbf{B}, \tag{11}$$

and

$$\mathcal{L}_2(\mathbf{A}, \mathbf{P}) = \mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P}. \tag{12}$$

The optimization problem represented by (9) – (12) is non-convex. Conventional solution technique for this problem involves the use of branch and bound global search [5]. However, the convergence rate is usually very slow since the BB's upper and lower bound are difficult to obtain and most estimations are conservative ones. It is the reason why, in this paper, evolutionary optimizers are used to solve the minimization problem.

4. Standard evolution strategies

Evolution strategies were first proposed by I. Rechenberg and H.–P. Schwefel as a set of heuristic optimizers to solve engineering problems [6]. The working principal is similar to that of a real-parameter genetic algorithm. Early ESs operate on real parameters and use only mutation operators. Recent developments have introduced recombination operators akin to crossover operators found in simple genetic algorithms. The following subsections describe these standard evolution strategies.

4.1 Two member ESs or $(1 \overset{+}{,} 1)$–ES

In two member ESs, one parent solution $\mathbf{k} \in \mathcal{R}^{n_k}$ is used to create one offspring solution $\mathbf{y} \in \mathcal{R}^{n_k}$ via a Gaussian mutation operator. There exists two different two member ESs. They differ in the way a solution is selected to become the parent for the next iteration. The one denoted by (1+1)–ES compares the parent solution with the offspring solution and the winner becomes the next parent. In (1,1) –ES the next parent solution is always the current offspring. The pseudo-code in Fig. 2 explains the working principal of a two member ES.

In Fig. 2, N(0, σ) corresponds to a vector created by a zero-mean normal distribution having a standard deviation σ. In practical implementations, (1+1)–ES and (1,1)–ES are deployed with static adaptation of σ by the so-called '1/5 success rule' [7]. This static adaptation rule computes a new value for the step size σ based on the number of successful mutations m_s over t trials.

```
1. Choose initial solution k and step size σ.
2. Create mutated solution: y = k + N(0, σ)
3. if (1+1)–ES then
4.   if f(y) < f(k), k = y endif
5. else
6.   k = y
7. endif
8. If termination criterion satisfied, exit, else goto step 2.
```

Figure 2 Standard $(1\overset{+}{,}1)$ –ES pseudo-code.

A successful mutation is one that produces a positive result in step 4 of Fig. 2. So the new step size σ' is

$$\sigma' = \begin{cases} \varsigma\sigma, & \text{if } m_s < 1/5; \\ \frac{1}{\varsigma}\sigma, & \text{if } m_s > 1/5; \\ \sigma, & \text{if } m_s = 1/5. \end{cases} \quad (13)$$

The constant $\zeta < 1$ in (13) is a heuristic factor that needs to be determined to suite a particular application.

4.2 Multi–member ESs or $(\mu\overset{+}{,}\lambda)$–ES

This is a population approach for the ES. In multi-member ESs, there are μ parent solutions $\mathbf{x}^{(i)}$, $i = 1, 2, \ldots, \mu$ used in the creation of λ offspring $\mathbf{y}^{(j)}$, $j = 1, 2, \ldots, \lambda$. Here again, the main difference between the (μ+λ)–ES and (μ,λ)–ES is in the selection of the new parents. With (μ+λ)–ES, the new parent solutions are the best solutions in a combined parent-offspring population whose size is μ+λ. Thus, (μ+λ)–ES is an elitist approach while the (μ,λ)–ES is a non elitist approach. Figure 3 gives the multi-member ES in pseudo-code.

1. Choose initial population P of solutions $\mathbf{k}^{(i)}$, i=1, 2, …, μ and step size σ.
2. Create λ mutated solutions: $\mathbf{y}^{(j)} = \mathbf{k}^{(i)} + \mathbf{N}(0, \sigma)$
3. if (μ+λ)–ES then
4. $\quad P' = \left(\bigcup_{j=1}^{\lambda}\{\mathbf{y}^{(j)}\}\right) \cup \left(\bigcup_{i=1}^{\mu}\{\mathbf{k}^{(i)}\}\right)$
5. else
6. $\quad P' = \left(\bigcup_{i=1}^{\lambda}\{\mathbf{y}^{(i)}\}\right)$
7. endif
8. Select μ best solutions from P' to become new parents
9. If termination criterion satisfied, exit, else goto step 2.

Figure 3 Standard $(\mu\overset{+}{,}\lambda)$ –ES pseudo-code.

To create the j-th offspring solution in step 2, a parent $\mathbf{k}^{(i)}$ is chosen at random from the parent population. It is obvious from step 6 and step 8 that $\lambda \geq \mu$.

4.2.1 Multi–member ESs with recombination

In multi-member ESs the offspring creation procedure involves the selection of a parent solution from a pool of μ potential candidates (see step 2, Fig. 3). Instead of a purely random approach, it is possible to apply a crossover-like recombination operator in the parent selection process. The application of this operator should come before the offspring creation step. There exists two commonly used recombination operators called *intermediate* and *discrete* recombination operator.

An intermediate recombination operator defines a parent solution for offspring creation as the average of a subset of parents in the current population, Thus, the parent produced by an intermediate recombination operator is

$$\hat{\mathbf{k}} = \frac{1}{\tau}\sum_{i=1}^{\tau\leq\mu}\mathbf{k}^{(i)}, \quad (14)$$

where $\hat{\mathbf{k}} = [\hat{k}_1\,\hat{k}_2 \ldots \hat{k}_{n_k}]^T$ is the new parent solution and $\mathbf{k}^{(i)} \in$ P. A discrete recombination operator chooses each element of a solution $\hat{\mathbf{k}}$ from one of the τ parent solutions at random. This is akin to the genetic algorithm's uniform crossover operator. Here, the newly formed solution will be the parent in the offspring creation procedure. Given an optimization problem with n_k decision variables, a subset of $\tau \leq \mu$ parent solutions and a discrete random variable r, a discrete recombination operator generates a parent $\hat{\mathbf{k}}$ by applying

$$\hat{k}_i = k_i^{(r)}, \quad i = 1, 2, \ldots, n_k, \quad r \in \{1, \ldots, \mu\}. \quad (15)$$

Finally a multi-member $(\mu\overset{+}{,}\lambda)$–ES with recombination operator is often denoted by $(\mu/\alpha\overset{+}{,}\lambda)$–ES where $\alpha \triangleq \tau_I$ means intermediate recombination using $\tau \leq \mu$ parents. When $\alpha \triangleq \tau_D$ it means that the recombination operator uses $\tau \leq \mu$ parents and is a discrete one.

4.2.2 Self-adaptation of step size σ

The so-called '1/5 success rule' for step size adaptation is a static update strategy. As shown by (13), the increase and decrease of the step size is based on the realized successful mutations. However, this is not a theoretical concept but empirical observations reported by Rechenberg [8].

Another adaptation approach is to let the step size evolve within an ES and it is called self-adaptation. The reason for self-adaptation is that most often the lower and upper bounds for optimal solution are not known. Furthermore, it may be necessary to obtain solutions with great precision [9]. There exist two basic self-adaptation strategies in use. The 'isotropic self-adaptation' strategy and the 'non-isotropic self-adaptation' strategy. In the former, there is one global step size σ for the creation of an offspring. A logarithmic update rule is used to evolve the global step size. Consequently, the step size update and offspring creation are described by

$$\begin{aligned} \sigma' &= \sigma\exp(m\mathrm{N}(0,1)), \\ \mathbf{y} &= \mathbf{k} + \sigma'\mathbf{N}(0,1), \end{aligned} \quad (16)$$

where σ' is the updated step size, $\mathbf{y}$ is an offspring, $\mathbf{k}$ is the parent solution chosen randomly or generated by a recombination operator. $\mathbf{N}(0,1)$ is $n_k \times 1$ normally distributed vector with zero-mean and standard deviation of one. In (16), the constant m is called the learning parameter. Its optimal value is not known. Most researchers reported that $m \propto 1/\sqrt{2n_k}$, where n_k is the number of decision variables, does provide interesting results [9, 10].

In non-isotropic self-adaptation, there is a different step size σ_i for each decision variable. Thus, each decision variable has its own mutation step size and n_k step sizes have to be updated accordingly. The step size updates and offspring creation are described by

$$\begin{aligned}\sigma'_i &= \sigma_i \exp(m\mathrm{N}(0,1) + m'\mathrm{N}_i(0,1)),\\ y_i &= k_i + \sigma'_i\,\mathrm{N}_i(0,1),\end{aligned} \tag{17}$$

where σ_i is the step size of the i-th decision variable. Similar to the isotropic self-adaptation strategy, m and m' are the heuristic learning parameters. Again, their optimal values are not known. In [10], the suggestion is to use $m \propto 1/\sqrt{2n_k}$ while setting $m' \propto 1/\sqrt{2\sqrt{n_k}}$.

4.2.3 Recombinant self-adapted ESs

For practical applications multi-member ESs is often deployed with a recombination operator and a self-adaptation strategy. This results in a set of multi-member recombinant self-adapted ESs. Figure 4 shows the $(\mu/\alpha \overset{+}{,} \lambda)$–ES with self-adaptation.

1. Choose initial population P of solutions $\mathbf{k}^{(i)}$, i=1, 2, …, μ.
2. if isotropic self-adaptation, choose initial step size σ, else choose n_k initial step size σ_i, endif
3. if intermediate recombination, use Eq. (14) to produce a parent $\hat{\mathbf{k}}$ else use Eq. (15) to produce a parent $\hat{\mathbf{k}}$ endif
4. if isotropic self-adaptation, use Eq. (16) to update step size and produce λ offspring else use (17) endif
5. if (μ+λ)–ES then
4. $\mathrm{P}' = \left(\bigcup_{j=1}^{\lambda}\{\mathbf{y}^{(j)}\}\right) \cup \left(\bigcup_{i=1}^{\mu}\{\mathbf{k}^{(i)}\}\right)$
5. else
6. $\mathrm{P}' = \left(\bigcup_{i=1}^{\mu}\{\mathbf{k}^{(i)}\}\right)$
7. endif
8. Select μ best solutions from P' to become new parents
9. If termination criterion satisfied, exit, else goto step 3.

Figure 4 Standard $(\mu/\alpha \overset{+}{,} \lambda)$–ES pseudo-code.

In Fig. 4, there are two possible recombination operators, two possible self-adaptation strategies and two different parent selection procedures. Thus, the illustrated pseudo-code plus corresponds to 8 different evolution strategies.

5. Structured controller design

In this section, 19 different ESs are applied to the design of a structured controller. These 19 ESs are:

- (1+1)–ES with static adaptation;
- (μ+λ)–ES with static adaptation, isotropic and non isotropic) self-adaptation;
- (μ/τ_I+λ)–ES with static adaptation, isotropic and non isotropic self-adaptation;
- (μ/τ_D +λ)–ES with static adaptation, isotropic and non isotropic self-adaptation;
- (μ,λ)–ES with static adaptation, isotropic and non isotropic self-adaptation;
- (μ/τ_I,λ)–ES with static adaptation, isotropic and non isotropic self-adaptation;
- (μ/τ_D,λ)–ES with static adaptation, isotropic and non isotropic self-adaptation.

The system model shown in Fig. 1 is used in this structured controller design. The reference model matrices, the plant model and the sensors model matrices are, according to Eqs (1), (2) and (3)

$$\left[\begin{array}{c|c}\mathbf{A}_r & \mathbf{B}_r\\\hline \mathbf{C}_r & \end{array}\right] = \left[\begin{array}{c|c}-0.4 & 1\\\hline 0.4 & \end{array}\right],\ \left[\begin{array}{c|c}\mathbf{A}_s & \mathbf{B}_s\\\hline \mathbf{C}_s & \mathbf{D}_s\end{array}\right] = \left[\begin{array}{c|c}\varnothing & \varnothing\\\hline \varnothing & 1\end{array}\right],$$

$$\left[\begin{array}{c|c}\mathbf{A}_p & \mathbf{B}_p\\\hline \mathbf{C}_p & \end{array}\right] = \left[\begin{array}{cccc|c}0 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ -1 & -4 & -6 & -4 & 1\\\hline 1 & 0 & 0 & 0 & \end{array}\right].$$

In order to match the output of the plant to the output of the reference model for a step input response, the weight error model matrices are chosen, according to Eq. (4) to be

$$\left[\begin{array}{c|c}\mathbf{A}_w & \mathbf{B}_w\\\hline \mathbf{C}_w & \end{array}\right] = \left[\begin{array}{c|c}-0.004 & 1\\\hline 1 & \end{array}\right].$$

Note that this weight error model is a stable approximation of the unit step function. The impulse response of the complete system is then close to the step response without the weight error model.

The controller model is a PID controller with a 100 rad/s low-pass filter. The matrices of this structured controller are

$$\left[\begin{array}{c|c}\mathbf{A}_c & \mathbf{B}_c\\\hline \mathbf{C}_c & \mathbf{D}_c\end{array}\right] = \left[\begin{array}{cc|c}0 & 1 & 0\\ 0 & -100 & 100\\\hline 0 & 0 & 0\end{array}\right] + k_1\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1 & 0 & 0\end{bmatrix} + k_2\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}0 & 1 & 0\end{bmatrix} + k_3\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}0 & -100 & 100\end{bmatrix}$$

and the controller gains are $\mathbf{k} = [k_i\ k_p\ k_d]^T$ where k_i, k_p and k_d are respectively the integral, the proportional and the derivative controller gains. Thus, in this problem instance, $n_k = 3$.

5.1 ES parameters

The ES optimizers are implemented in order to solve an instance of the control problem represented by the non convex optimization problem of Eqs (9) – (12). The control system model parameters are given in section 5. Here, Table I is a summary of the important ES parameters. In this work all 19 ESs use the same set of parameters.

Table I. Summary of ES parameters

$\overline{\mathbf{x}}$	$\underline{\mathbf{x}}$	$\breve{\sigma}$	Nb. eval.
$[10\ 10\ 10]^T$	$[0\ 0\ 0]^T$	1 or $[1\ 1\ 1]^T$	5000

μ	λ	τ	ζ	Nb. trials
20	100	10	0.817	10

In Table I, the $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$ are each decision variable's upper and lower bounds. Recall that the decision variables are the k_i, k_p and k_d controller gains. The $\breve{\sigma}$ parameter is the initial step size. For the isotropic self-adaptation strategy, $\breve{\sigma}$ is a scalar. For the non isotropic case, $\breve{\sigma}$ is a 3×1 vector. In this work, the ESs have $\mu = 20$ parent solutions and $\lambda = 100$ offspring solutions. The ratio $\lambda/\mu \approx 5$ and the static adaptation factor $\zeta = 0.817$ are often suggested in the ES literature [6, 7, 11]. Finally, each of the 19 ESs must evaluate Eq. (9) 5000 times and 10 trials are conducted for each ES.

5.2 Experimental results

To obtain a fair comparison, a branch and bound (BB) global search algorithm is used to find the optimal controller gains. The BB algorithm is described in [12]. Its took over 170 hours of computing time on a 1.1 GHz Athlon personal computer using the $\overline{\mathbf{x}}$ and $\underline{\mathbf{x}}$ given in Table I and the MATLAB programming environment. In contrast, each ES trial took 21 minutes to complete using the same computer and the same programming environment. The following tables show the results produced by the ESs. They are compared to the results produced by the BB algorithm. Table II gives the label corresponding to each ES. In Table III, IV and V, the column denoted *f* indicates the objective value or solution quality obtained by optimizing Eq. (9).

Table II. ES identification

	ES		ES
A	(1+1)	K	(μ,λ)
B	(μ+λ)	L	(μ,λ), isotropic sa
C	(μ+λ), isotropic sa	M	(μ,λ), non isotropic sa
D	(μ+λ), non isotropic sa	N	$(\mu/\tau_I,\lambda)$
E	$(\mu/\tau_I+\lambda)$	O	$(\mu/\tau_I,\lambda)$, isotropic sa
F	$(\mu/\tau_I+\lambda)$, isotropic sa	P	$(\mu/\tau_I,\lambda)$, non isotropic
G	$(\mu/\tau_I+\lambda)$, non isotropic sa	Q	$(\mu/\tau_D,\lambda)$
H	$(\mu/\tau_D+\lambda)$	R	$(\mu/\tau_D,\lambda)$, isotropic sa
I	$(\mu/\tau_D+\lambda)$, isotropic sa	S	$(\mu/\tau_D,\lambda)$, non isotropic sa
J	$(\mu/\tau_D+\lambda)$, non isotropic sa		

Table III. BB results after 170 hours.

	k_i	k_p	k_d	f
BB	0.3955	1.2256	2.1582	0.049

Table IV and V show that evolution strategy E, F, I, N, O and R failed to compete against the BB algorithm. Their mean objective value exceeded the one obtained by the BB algorithm. The failed ESs are marked by the symbol † in the tables. All other ESs were able to obtain a better design than the BB algorithm.

Table IV. ES: Results on *f*.

ES	min	max	mean	std. dev.
A	0.0489198	0.0489261	0.048923	1.761×10^{-6}
B	0.0489222	0.0489246	0.0489231	6.852×10^{-7}
C	0.0489247	0.0494162	0.0489938	1.5138×10^{-4}
D	0.0489234	0.0489954	0.0489363	2.4059×10^{-4}
E†	0.061712	0.164843	0.101286	0.0341494
F†	0.0489635	0.417083	0.0964839	0.114816
G	0.0489226	0.048924	0.0489232	4.3506×10^{-7}
H	0.0489224	0.0489245	0.0489232	7.9812×10^{-7}
I†	0.0489249	0.525461	0.113088	0.146554
J	0.0489221	0.0489232	0.0489227	4.0704×10^{-7}

Table V. ES: Results on *f*. (cont'd)

ES	min	max	mean	std. dev.
K	0. 0489231	0. 0489236	0.0489234	1.767×10^{-7}
L	0.0489198	0.0489241	0.0489229	1.255×10^{-6}
M	0.0489223	0.0489241	0.048923	5.905×10^{-7}
N†	0.072112	0.182639	0.137556	0.0310818
O†	0.0491401	0.231803	0.111954	0.0721626
P	0.0489215	0.0489239	0.0489228	6.6313×10^{-7}
Q	0.0489217	0.0489236	0.0489227	5.913×10^{-7}
R†	0.0492277	0.388956	0.181981	0.124669
S	0.0489216	0.0489252	0.0489233	1.1659×10^{-7}

Table VI. ES: Number of evaluations to reach within 0.5% of BB result

ES	min	max	mean	std. dev.
A	913	1425	1145	117
B	1126	1685	1355	216
C	–	–	–	–
D	1066	4250	2040	1055
E	–	–	–	–
F	–	–	–	–
G	1015	1799	1461	238
H	1021	1655	1378	192
I	–	–	–	–
J	943	1204	1078	98

Table VII. ES: Number of evaluations to reach within 0.01% of BB result (cont'd)

ES	min	max	mean	std. dev.
K	1163	2127	1629	344
L	1170	1893	1513	203
M	1112	2096	1607	275
N	–	–	–	–
O	–	–	–	–
P	1460	2255	1825	246
Q	1277	1923	1521	205
R	–	–	–	–
S	841	1655	1086	220

Next, the number of evaluations needed to attain 0.5% of the BB's final objective value were found. To appreciate the convergence rate of the ESs, a strict elimination procedure was used. An ES is rejected if, anytime during the 10 trial runs, it produces a final minimum objective value that is larger than the BB's. The convergence rate of the ESs is shown in Table VI and VII.

From Table VI and VII, one more ES is eliminated (the (μ+λ) with isotropic self-adaptation). Notice that almost all remaining ESs were able to surpass the BB algorithm within 2000 evaluations. Finally, Fig. 5 and 6 show the variability in terms of objective values and number of evaluations for the structured controller design problem.

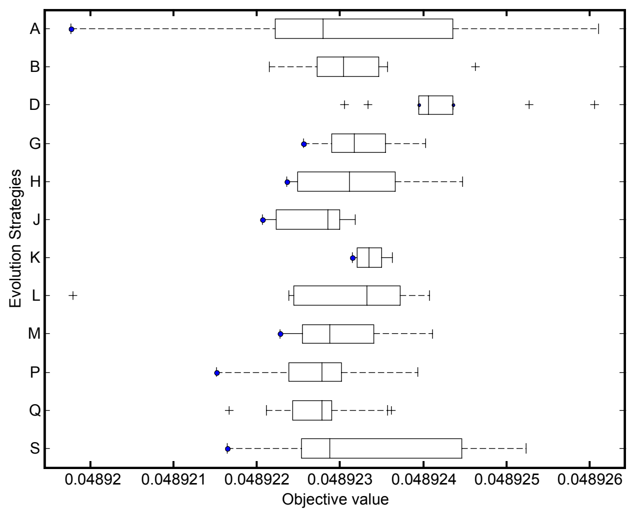

Figure 5 Boxplot showing the variability on the objective value.

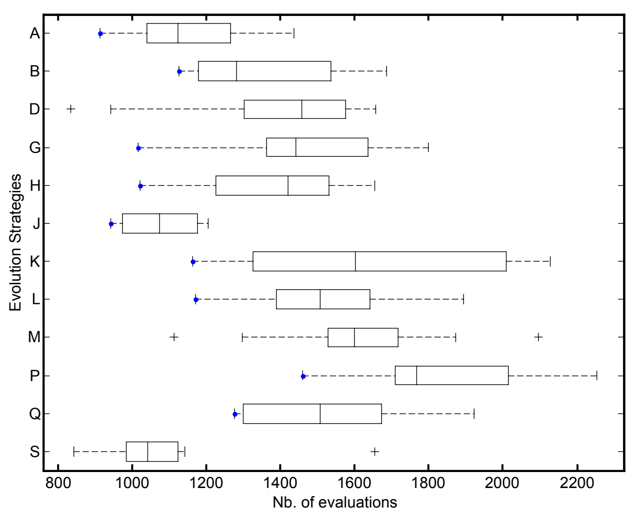

Figure 6 Boxplot showing the variability on the nb. of evaluations.

For the objective values, all remaining ESs performed remarkably well. Their variability, shown in Fig. 5, is negligible (the x-axis has a very small scale). For the number of evaluations, the evolution strategies J and S both have less variability and a smaller median. For the structured controller design problem, evolution strategies are capable of finding the global optimal solution. From the results of this work, the following observations are drawn:

- Elitist ES performed better than a non elitist ES;
- Intermediate recombination operator seems to be inadequate for this problem;
- Non isotropic self-adaptation is better than isotropic self-adaptation;
- (1+1)-ES with static adaptation is comparable with other ES but with a larger variability in solution quality.

6. Conclusion

A set of standard ESs has been used to find the optimal tuning of a structured controller design problem. The results are compared to those found by a branch and bound algorithm. Even without any specialized extension, most ESs were able to find the optimal solution of the resulting non-convex optimization problem. Furthermore, the ESs have almost no overhead and require considerably less computation time (21 min) compared to the BB algorithm (170 h).

References

[1] J. H. Lee, Nonlinear programming approach to biaffine matrix inequality problems in multiobjective and structured control, *Proc. American Control Conference*, 1999, 1817-1821.

[2] M. Chilali, C. Scherer & P. Gahinet, Multiobjective output-feedback control via LMI optimization, *IEEE Transactions on Automatic Control*, *42*(7), 1997, 896-911.

[3] T, Iwasaki, M. A. Rotea, An alternative to the d-k iteration, *Proc. American Control Conference*, 1994, 53-57.

[4] R. E. Skelton, J. C. Geromel & C. C. de Souza, LMI numerical solution for output feedback stabilization, *Proc. IEEE Conference on Decision and Control*, 1995, 40-44.

[5] V. Balakrishnan, S. P. Boyd, Global optimization in control system analysis and design, *Control and Dynamic Systems*, *53*, 1992, 1-55.

[6] H.–P. Schwefel, *Evolution and optimum seeking* (New York, John Wiley & Sons, 1995).

[7] T. Bäck, D. Fogel & Z. Michalewicz (Eds), *Handbook of evolutionary computation* (New York, Oxford University Press, 1997).

[8] R. Hinterding, Z. Michalewicz & A. E. Eiben, Adaptation in evolutionary computation: A survey, *IEEE Int. Conference on Evolutionary Computation*, 1997, 65-69.

[9] K, Deb, H.–G. Beyer, Self-adaptive genetic algorithms with simulated binary crossover, *Evolutionary Computation Journal*, *9*(2), 2001, 195-219.

[10] T. P. Runarsson, R. Sarker & M. T. Jonsson, Constrained nonlinear integer programming, self-adaptation and evolution strategies, *International Journal of Knowledge-Based Intelligent Engineering Systems*, *4*(3), 2000, 164-171.

[11] T. Bäck, An overview of parameter control methods by self-adaptation in evolutionary algorithms. *Fundamenta Informaticae*, *45*(1-4), 1998, 51–66.

[12] P. Bigras, T. Wong, K Khayati, Structured optimal controller design using BMI formulation, *Multiconference on Computational Engineering in Systems Applications*, 2003. (Accepted)