Tạp chí Tin học và Điều khiển học, T. 15, S. 3 (1999), 50-54

H_∞-OPTIMAL CONTROLLER DESIGN USING GENETIC ALGORITHMS

VU NGOC PHAN

Abstract. The goal of the present paper is to verify the possibility of genetic algorithms for solving the H_∞-optimal controller design problem. After formulating the design problem in Section 2 and repeating the background of genetic algorithms in Section 3, some simulation results using MATLAB 5.2 will be explained in Section 4. In Section 5 the method will be shortly discussed.

1. INTRODUCTION

The H_∞-optimal control was developed at the beginning 1980s when many researchers on the field of control system theory paid their attention to MIMO systems more than to SISO one [1]. The H_∞-optimal control can be considered as an extension of the well-known least square method. Moreover, the H_∞-optimal control has had a worthy contribution to the development of the robust control theory, especially robust controller design in the frequency domain [2]. In the last decade there was many investigations to this issue. However, in the fact there has been no realizable success in solving the H_∞-optimal problem yet because of many difficulties. Firstly, the H_∞-optimal problem belong to MINMAX issue, which is well-known very difficult. Secondly, the description of plants to be controlled in the frequency domain has many advantages, but it is not easy to check the system stability especially if there are parameter and/or dynamic model uncertainties [5, 6, 7]. When there is an increase in the number of plant input and/or output the problem becomes terribly difficult.

It is a need to find an adequate algorithm for solving the H_∞-optimal problem. The purpose of this paper is to show through a simulation example that the genetic algorithms may be an useful tool for the considered issue. In the following section the design problem based on the standard MIMO control configuration is shortly explained. Then in Section 3 the fundamental of genetic algorithms is repeated including some modifications. It is to note here that the genetic algorithms are essentially heuristic. The convergence of algorithm could not be proved in the paper.

2. H_∞-OPTIMAL CONTROLLER DESIGN PROBLEM

Figure 1 shows the standard configuration of MIMO control systems. Here $u \in \mathbb{R}^n$, $r \in \mathbb{R}^m$ and $y \in \mathbb{R}^m$ are input vector, reference input vector and output vector, respectively. The transfer function matrices $P(s)$ and $C(s)$ are the controlled plant model and controller to be designed, respectively.

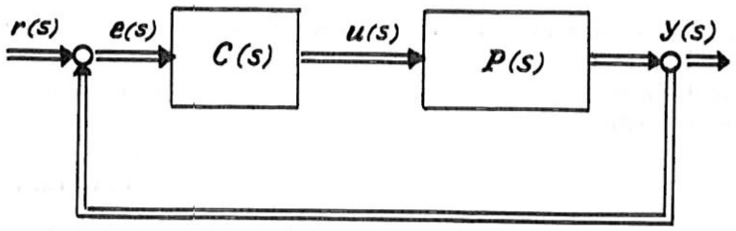


Fig. 1. MIMO control system

It follows from Fig. 1

$$y(s) = P(s)C(s)[I + PC]^{-1} r(s) = H(s)r(S), \tag{2.1}$$
$$e(s) = r(s) - y(s) = [I + PC]^{-1} r(s) = E(s)r(s). \tag{2.2}$$

Assuming that the structure of $C(s)$ is known and only its parameters must be determined. Let $C(p,s)$ denote such a controller where p is the parameter vector with an upper and a lower limit. That is

$$a \le p \le b\,. \tag{2.3}$$

The goal of the design problem is to find a parameter vector p such as the error $e(s)$ becomes small due to an objective function. In general, this requirement can not be satisfied at every frequency. As well known, the limit expressions

$$\lim_{s\to\infty} PC = 0 \text{ and } \lim_{s\to\infty} E(s) = 1\,. \tag{2.4}$$

are satisfied if PC is a proper transfer function matrix [3]. The objective function for minimizing $e(s)$ may be chosen as

$$\|e(j\omega)\|_2^2 = \frac{1}{2\pi}\int_{-\infty}^{+\infty} e^*(j\omega)e(j\omega)\,d\omega \tag{2.5}$$

or after putting $e(s)$ from (2.2) in (2.5)

$$\|e(j\omega)\|_2^2 = \frac{1}{2\pi}\int_{-\infty}^{+\infty} r^*(j\omega)E^*(j\omega)E(j\omega)r(j\omega)\,d\omega\,. \tag{2.6}$$

In the point of fact, only reference input signals with finite power spectrum are of great interest [3]. Such signals may be expressed by

$$r(s) = W(s)r_0\,, \tag{2.7}$$

where $W(s)$ is a transfer function matrix that has at least a pole in the right haft-plane and r_0 is a constant vector. From (2.6) and (2.7) it follows

$$\|e(j\omega)\|_2^2 \le \|E(j\omega)W(j\omega)\|_\infty^2\| \cdot \|r_0\|_2^2\,. \tag{2.8}$$

Because $\|r_0\|_2^2$ does not affect the optimality of (2.6) it is enough to consider the objective function

$$\min_p \|e(j\omega)\|_2 \cong \min_p \|E(j\omega)W(j\omega)\|_\infty \tag{2.9}$$

or in similarity

$$\min_p \sup_\omega \bar{\sigma}(E(j\omega)W(j\omega))\,. \tag{2.10}$$

The solution of (2.10) is called H_∞-optimal. The problem (2.10) is so difficult that there is no practically satisfied way for solving it [1]. In the following section the genetic algorithms are considered and modified in order to apply to this task.

3. GENETIC ALGORITHMS

The basic of genetic algorithms are explained in [4]. It is an extension of general-purpose optimization method for solving search problems. The method was developed based on the assertion that the evolution of living bodies underlying the principles of natural selection process is the best way to the optimality. Naturally, the natural evolution process is so complicated that it is impossible to simulate completely by a mathematical algorithm. The genetic algorithm used in this paper consists of following steps.

Step 1. Initialization. In this step an initial population is created, the fitness function is defined, some control parameters (reproduction rate KRED, crossover rate KCRO, mutation rate KMU) of simulation program can be entered through dialog questions.

Step 2. Reproducing new population. In this step a random number between 0 and the total fitness is generated. Then the first population member whose fitness is greater or equal to the generated random number returns to the population once more as a new member. The reproduction rate can be adjusted by a control parameter KRED (see function REPROD).

Step 3. Crossover. In this step two individuals of the population are randomly selected as parents. The parents change randomly one or more of their chromosomes to each other. The crossover rate KCRO determines how often the crossover operation occurs.

Step 4. Mutation. In this step one or some individuals of the population are randomly selected. After that one or some chromosomes of the selected individual are randomly changed. The mutation rate KMU decided how often the mutation takes place.

Step 5. Reduction. After the reproduction process there is an increase in the number of individuals. It is impossible to deal with a too large population. Hence, it is a need to eliminate some individuals before next evolution step. In the simulation program of this paper the population size is kept constant. Thus, a number of the worst individuals must be eliminated so that the population size does not grow.

The step from 2 to 5 can be repeated according to the answer to a dialog question. After every evolution step all the data of ten best individuals is displayed, so that an user can see and decide whether he continues or stops the evolution process.

4. SIMULATION RESULT

As well known, the H_∞-optimal solution does not ensure that the closed-loop system is stable. For overcoming that obstacle it is necessary to introduce some constraints to the designing procedure. The easy way may be that all the poles of the closed-loop system are obligated to lie in a restricted zone of the left-haft plane, e.g.

$$\Pi := \{z \mid z \in C;\ z = -r_0 + r\cos(\alpha) + jr\sin(\alpha);\ r_0 = a,\ 0 < r < b;\ a, b \text{ positive real number},\ 90^\circ < \alpha < 270^\circ\} \tag{4.1}$$

The result presented below is obtained by running Programme HDESIGN. The plant is a two-input two-output system that has two unstable poles at 0.5 and 3.

$$P(s) = \begin{pmatrix} \dfrac{1.2}{s+2.5} & \dfrac{1.5}{s-3} \\ \dfrac{-2}{s+1.5} & \dfrac{2.6}{s-0.5} \end{pmatrix} \tag{4.2}$$

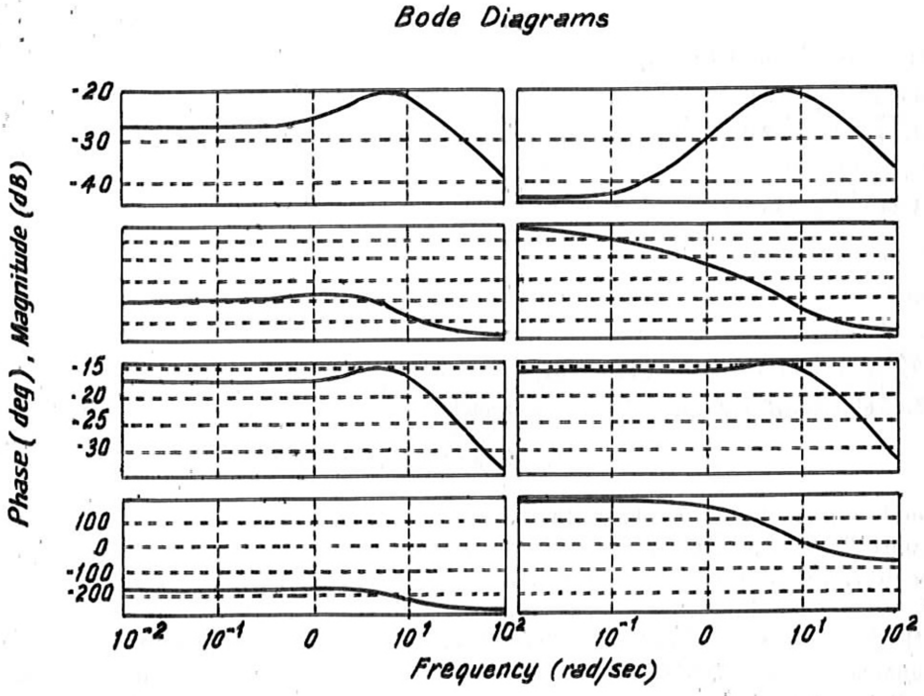
It is necessary to remark here that the author of this paper has not find out any meaningful criterion for automatically stopping the evolution procedure. As mentioned in Section 3, we can stop running the simulation programme when the momentary result is acceptable. Therefore, the simulation is stopped after 15 evolution steps. The transfer function matrix of the closed-loop system is expressed in original MATLAB 5.2 format.

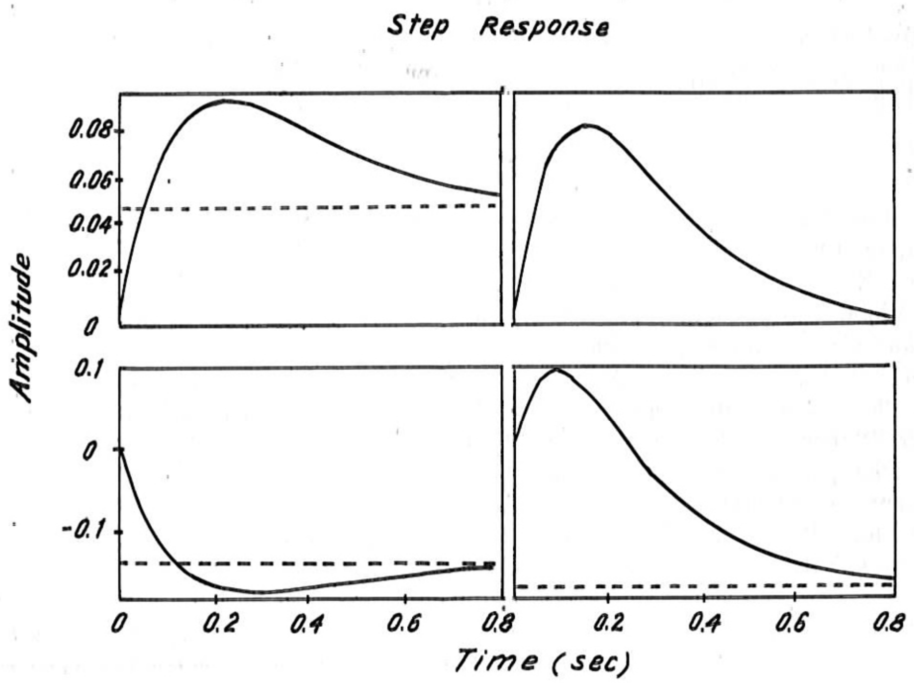
Closed-loop system

Transfer function from input 1 to output

$$\#1: \quad \frac{1.2\,s\text{^}3 + 19.62\,s\text{^}2 + 92.45\,s + 94.71}{s\text{^}4 + 25.92\,s\text{^}3 + 256.4\,s\text{^}2 + 1154\,s + 2010}$$

$$\#2: \quad \frac{-2\,s\text{^}3 - 33.21\,s\text{^}2 - 176.9\,s - 280.2}{s\text{^}4 + 25.92\,s\text{^}3 + 256.4\,s\text{^}2 + 1154\,s + 2010}$$





Transfer function from input 2 to output

$$\#1: \frac{1.5\, s^{\wedge}3 + 18.78\, s^{\wedge}2 + 62.1\, s - 13.17}{s^{\wedge}4 + 25.92\, s^{\wedge}3 + 256.4\, s^{\wedge}2 + 1154\, s + 2010}$$

$$\#2: \frac{2.6\, s^{\wedge}3 + 17.99\, s^{\wedge}2 - 17.41\, s - 338.5}{s^{\wedge}4 + 25.92\, s^{\wedge}3 + 256.4\, s^{\wedge}2 + 1154\, s + 2010}$$

Controller

Transfer function from input 1 to output

$$\#1: \frac{7.421\, s\hat{}3 + 203.1\, s\hat{}2 + 1821\, s + 3226}{s\hat{}4 + 53.34\, s\hat{}3 + 838.1\, s\hat{}2 + 4371\, s + 1.011e004}$$

$$\#2: \frac{-17.53\, s\hat{}3 - 333.5\, s\hat{}2 - 757.1\, s - 100.4}{s\hat{}4 + 53.34\, s\hat{}3 + 838.1\, s\hat{}2 + 4371\, s + 1.011e004}$$

Transfer function from input 2 to output

$$\#1: \frac{7.667\, s\hat{}3 + 284.1\, s\hat{}2 + 1311\, s + 1761}{s\hat{}4 + 53.34\, s\hat{}3 + 838.1\, s\hat{}2 + 4371\, s + 1.011e004}$$

$$\#2: \frac{2.326\, s\hat{}3 + 44.63\, s\hat{}2 + 461\, s - 802.7}{s\hat{}4 + 53.34\, s\hat{}3 + 838.1\, s\hat{}2 + 4371\, s + 1.011e004}$$

5. CONCLUSION AND REMARK

From simulation results it was shown that the genetic algorithms may used easily to solve the H_∞-optimal controller design problem. Genetic algorithms can be very flexible constructed as well as modified for different special problem. However, the accuracy of computing seems to cause some troubles. When a pole is only a few percent different from a zero it may be thought that they were the same. If they were the same and the difference occur from computing errors, the dynamic of controller is unnecessary high. In that case it seems to be better to cancel them and then the system stability needs checking again.

The original paper has an appendix containing the simulation program HDESIGN. Because of a page restriction of the journal the appendix was not published. The reader who is interesting in the issue could contact with the author by e-mail address *vnphan@fpt.vn*.

REFERENCES

[1] J. Doyle, Robust and optimal control, *Proceeding of the 35 Conference on Decision and Control*, Kobe, Japan, 1996.

[2] M. Morari, E. Zafiriou, *Robust Process Control*, Prentice Hall. N.J., 1986.

[3] Vu Ngoc Phan, Robust control of MIMO systems with input uncertainties, *Journal of Computer Science and Cybernetics* **12** (3) (1996).

[4] Chin-Teng Lin, C. S. George Lee, *Neural Fuzzy Systems*, Prentice-Hall International, Inc., 1996.

[5] Vu Ngoc Phan, Disturbance rejection of MIMO control systems with uncertainties, *Journal of Computer Science and Cybernetics* **13** (3) (1997).

[6] Vu Ngoc Phan, Automatic systems: uncertainties and control, *Journal of Computer Science and Cybernetics* **13** (4) (1997).

[7] Vu Ngoc Phan, Robust stability analysis of IMC systems, *Journal of Computer Science and Cybernetics* **14** (1) (1998).

Received October 26, 1998
Revised April 15, 1999

Institute of Information Technology