

Robust Stability Analysis of Discrete-Time Systems Using Genetic Algorithms

M. Sami Fadali, Yongmian Zhang , Sushil J. Louis

Abstract— We reduce stability robustness analysis for linear, time-invariant, discrete-time systems to a search problem and attack the problem using genetic algorithms. We describe the problem framework and the modifications that needed to be made to the canonical genetic algorithm for successful application to robustness analysis. Our results show that genetic algorithms can successfully test a sufficient condition for instability in uncertain linear systems with nonlinear polynomial structures. Three illustrative examples demonstrate the new approach.

Keywords— Robust Stability Analysis, Genetic Algorithms

I. INTRODUCTION

Although robust stability and control has been an active area of research since the 1970's [1], [2], [3], the Kharitonov approach to robust stability analysis is of more recent vintage [4]. Most existing work on robust stability analysis is based on relatively simple uncertainty structures and for systems with more complex uncertainty structures, these approaches are no longer feasible. Kharitonov's theorem for example [5], only applies to continuous-time systems with interval polynomials. Although the edge theorem relaxes Kharitonov's assumptions, the time complexity for this approach for a fixed order polynomial is $O(n2^{n-1})$, where n is the number of polytope edges [6]. Thus, it is not computationally tractable for even modestly large numbers of uncertain parameters. Zadeh provided an important tool known as the mapping theorem for testing the stability of polynomials with multilinear coefficients, but its time complexity for a fixed order polynomial is even worse ($O(2^{2n-1})$) [7].

Although there is no discrete analog of Kharitonov's theorem, robust Schur stability analysis has been widely investigated [1], [4], [1], [8], [9]. Unfortunately, the approaches in the literature are not *in general* applicable to polynomials with nonlinear uncertainty structures. The results can only be applied to non-linear uncertainty structures if overbounding is used, and provide a sufficient test for stability [4], [10]. In general, the main approach for handling nonlinear uncertainty structures is by dense gridding of the uncertainty domain. In special cases, alternative approaches, such as the construction of the value sets using tree structured decomposition [10], are feasible. The questions of stability and stabilization of complex uncertain systems remain among the most important issues in control engineering today.

Dr. Fadali and Yongmian Zhang are with the Dept. of Electrical Engineering, University of Nevada, Reno, NV 89557.

Dr. Louis is with the Dept. of Computer Science, University of Nevada, Reno, NV 89557.

Recently, genetic algorithms (GAs) have been widely applied to effectively solve difficult optimization problems [11], [12], [13]. These algorithms have proven to be efficient for complex problems which are not computationally tractable using other approaches. GAs have attractive features for robust control problems because they do not require linearity, continuity, or other restrictions as in classical approaches. Furthermore, genetic algorithms offer the attraction that all parts of the feasible space are potentially available for exploration. This enhances the robustness properties of genetic search and the results obtainable for problems under investigation. Some researchers presented encouraging results by applying GAs to control parameter estimation [14], [15], [16], optimal control [17], [18] and robust controller design [19], [20], [21], [22], [23]. To our knowledge, genetic algorithms have not been utilized for robust stability analysis of discrete-time uncertain systems. In this paper, we investigate the applicability and adaptability of GAs in providing a sufficient test of *instability* for such systems. We believe our instability test complements the overbounding stability test and provides a useful tool for investigating robustness.

The remainder of the paper is organized as follows. In Section 2 we give a framework for robust stability analysis of discrete-time systems. Section 3 describes the canonical genetic algorithm while section 4 describes our modifications to the canonical genetic algorithm for application to robust stability analysis. We present computational results and analysis in Section 5. Conclusions and directions for future studies are presented in Section 6.

II. ROBUSTNESS OF DISCRETE-TIME SYSTEM

Consider a linear time invariant discrete-time system. The stability of the system can be determined by examining its characteristic equation. The characteristic equation may be written as:

$$P(z, \mathbf{q}) = \{p(z, \mathbf{q}) = \sum_{i=0}^n a_i(\mathbf{q})z^i, \mathbf{q} \in \mathbf{Q}\} = 0 \quad (1)$$

where $a_i(\mathbf{q})$, $i = 0, 1, \dots, n$ are functions of the uncertain plant parameter vector \mathbf{q} , and $\mathbf{q} \in \mathbf{Q} \subset R_n$. The coefficients a_i may be interval, affine, multilinear or even exponentially dependent on the uncertain plant parameter vector \mathbf{q} . When the coefficients are nonlinear, traditional methods fail, and overbounding or gridding are our only options. Unfortunately, overbounding gives conservative stability results while gridding is only feasible for a small number of parameters.

The stability boundary for Equation (1) is $|z| = 1$. The stability test is to determine if there exists a system pole outside the unit circle. Using the mapping $z' \rightarrow z^{-1}$, the instability region becomes the closed unit disc \bar{U} , and therefore, system instability testing reduces to searching for a pole inside or on the unit circle (see Figure 1).

If the polynomial coefficients are continuous functions of the uncertain parameters and no degree dropping occurs, the system instability test can be simplified to searching for a root on the unit circle and the mapping $z' \rightarrow z^{-1}$ need not be used. This follows from the well known boundary crossing theorem [10].

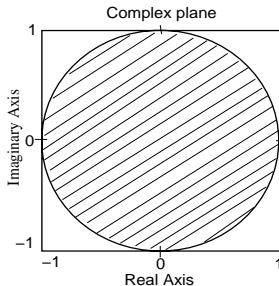


Fig. 1. Roots of $P(z', \mathbf{q})$ in the unit circle

Searching for a root inside or on the unit circle can be formulated as minimizing the cost function

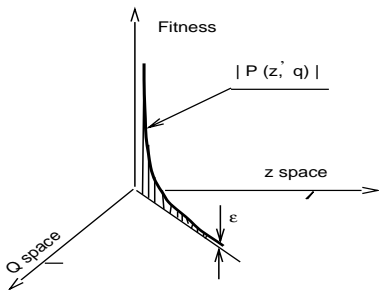


Fig. 2. Minimization of a cost function

$$F = |P(z', \mathbf{q})|, \quad z' \in \bar{U} \text{ and } \mathbf{q} \in \mathbf{Q} \quad (2)$$

with tolerance ϵ as shown in Figure 2 and where \bar{U} denotes the closed unit disc and \mathbf{Q} is the set of feasible parameter vectors. The robust stability problem can now be stated as follows: Given a family of characteristic functions P associated with uncertain physical parameters \mathbf{q} , search the space $\bar{U} \times \mathbf{Q}$ for a minimal value of $|P(z', \mathbf{q})|$, $\forall z' \in \bar{U}$ and $\forall \mathbf{q} \in \mathbf{Q}$. If \exists a "zero" of $|P(z', \mathbf{q})|$, then the system is unstable. If the polynomial is continuous and no degree dropping occurs, we search for the roots on the unit circle. If $P(Re^{j\theta}, \mathbf{q}) \neq 0$, $\forall \mathbf{q} \in \mathbf{Q}$, $\theta \in [0, \pi]$ and $R = 1$, then further search inside the unit circle will be necessary. Thus there are two main possibilities:

1. If we find a "zero" on or inside the unit circle, we have *sufficient* evidence to conclude that the system is unstable
2. Otherwise, if we do not find a "zero" there are two further possibilities.

- (a) The system is unstable but the genetic algorithm cannot find a "zero," or
- (b) The system is stable

We cannot distinguish between 2a and 2b and thus our approach can only provide a *sufficient* condition for instability, not a necessary condition. Specifically, not finding a "zero" of the polynomial does not provide any information about system stability.

A key issue that arises in this approach is the extremely large search space. In such a large search space, exhaustive search methods take unacceptably long while robust search algorithms provide a promising alternative. Genetic algorithms were designed to robustly and efficiently search large, nonlinear search spaces where traditional optimization techniques are not feasible. GAs are particularly attractive because they do not require the coefficients of the characteristic polynomial to be linear and we thus choose to use GAs as the search method for this paper.

III. GENETIC ALGORITHMS

GAs are stochastic, parallel search algorithms based on the mechanics of natural selection and the process of evolution [24], [12]. GAs were designed to efficiently search large, nonlinear spaces where expert knowledge is lacking or difficult to encode and where traditional optimization techniques fail. GAs perform a multi-directional search by maintaining a population of potential solutions usually encoded as bit strings and encourage information formation and exchange between these solutions. A population is modified by the probabilistic application of the genetic operators from one generation to the next. Whenever some individuals in the population exhibit better than average performance, the genetic features of these individuals will be copied with high probability to the next generation.

Evaluation of each string which encodes a candidate solution is based on a fitness function that is problem dependent. Selection is done on the basis of relative fitness and it probabilistically culls solutions from the population that have relatively low fitness. Recombination consists of mutation and crossover. Mutation insures against the permanent loss of genetic material during the selection process and with low probability flips a bit in the genotype. Crossover is a structured yet stochastic operator that allows information exchange between candidate solutions. Figure 3 shows that one point crossover is implemented by choosing a random point in the selected pair of strings and exchanging complementary substrings defined by the chosen point. Conceptually, GAs work with a rich population

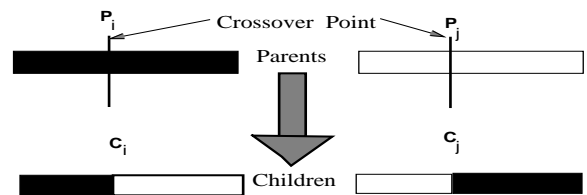


Fig. 3. One point crossover

and simultaneously climb many peaks in parallel during

search processing. This significantly reduces the probability of getting trapped at a local minimum. They are thus considered more robust and more broadly applicable than other similar search algorithms.

IV. GA ENCODING FOR ROBUST STABILITY ANALYSIS

In the general case, the closed-loop characteristic polynomials is defined in parametric form as

$$P(z, \mathbf{q}, \mathbf{k}) = \sum_{i=0}^n a_i(\mathbf{q}, \mathbf{k}) z^i \quad (3)$$

The controller parameter vector \mathbf{k} and the operating domain \mathbf{Q} are given. In such a polynomial family, the coefficient $a_i(\mathbf{q}, \mathbf{k})$ include any continuous *nonlinear* functions, in which we are particularly interested in coefficients with *nonlinear dependency*. The roots of this polynomial are complex and equal to $|z'|e^{j\theta}$ and the range of \mathbf{q} is defined by its pre-specified minimum and maximum values $[\mathbf{q}^-, \mathbf{q}^+]$. The objective of the genetic algorithm is to find:

$$f(z, q) = \text{Min}[|P(z', \mathbf{Q})|], \quad q \in \mathbf{Q}, \quad z \in \bar{U} \quad (4)$$

where

$$\begin{aligned} \mathbf{Q} &\subset R^n \text{ and } \bar{U} \subset R^2 \\ \mathbf{Q} &= \{\mathbf{q} : \mathbf{q} = [q_i], q_i^- \leq q_i \leq q_i^+, \forall i = 1, \dots, n\} \\ \bar{U} &= \{z' : |z'| \leq 1\} \end{aligned}$$

We exploit the fact that for real coefficients, roots are complex conjugate pairs or real. Hence, the search space can be restricted to the upper half of the unit circle shown in Figure 1.

A. Representation

We use a small margin of tolerance ϵ to represent a true numeric zero. ϵ can be the computer's floating-point precision, or double precision, since high precision is required in our problem. Additionally, the uncertain polynomial in real applications may contain numerous parameters. In such uncertain parameter optimization problems, a long bit string is required in order to extend the precision and represent the entire range of each parameter. Thus the binary chromosome representation traditionally used in GAs shows some deficiencies since a long chromosome usually results in poor performance for GAs.

In order to overcome the drawback of the binary representation, we coded the GA chromosome as a vector of real numbers with double precision. Each parameter q_i , $i = 1, \dots, n$, is initialized within the pre-specified domain $[q_i^-, q_i^+]$. The magnitude of the root z' is initially selected to be within $[0, 1]$, while the angle θ is in the interval $[0, \pi]$. Compared with binary representation, real numbers are better capable of representing our desired domain. The precision only depends on the machine we use and the accuracy depends on the user's requirements. We define the chromosome for the uncertain polynomial $p(z', \mathbf{q})$ as:

$$C_i^t = \langle q_1, \dots, q_k, \dots, q_n, z, \theta \rangle, \quad 1 \leq k \leq n \quad (5)$$

$$C_i^{t+1} = \langle q_1, \dots, q'_k, \dots, q_n, z', \theta' \rangle, \quad k \in (1, \dots, n) \quad (6)$$

where C_i^t is the chromosome of population i in t th generation. C_i^{t+1} is a new chromosome for the next generation after genetic selection. The q'_k , z' and θ' in C_i^{t+1} satisfy

$$q'_k = \begin{cases} q_k + \Delta(t, q_k - q_k^+) : \text{flip}\{0; 1\} = 0 \\ q_k - \Delta(t, q_k - q_k^-) : \text{flip}\{0; 1\} = 1 \end{cases} \quad (7)$$

$$z' = \begin{cases} z + \Delta(t, z - 1) & : \text{flip}\{0; 1\} = 0 \\ z - \Delta(t, z - 0) & : \text{flip}\{0; 1\} = 1 \end{cases} \quad (8)$$

$$\theta' = \begin{cases} \theta + \Delta(t, \theta - \pi) & : \text{flip}\{0; 1\} = 0 \\ \theta - \Delta(t, \theta - 0) & : \text{flip}\{0; 1\} = 1 \end{cases} \quad (9)$$

B. Tuning Genetic Algorithms

We assume that the polynomial $P(z', \mathbf{q})$ contains nonlinear coefficients, and may therefore have numerous finite local extrema. To enhance the GAs ability to efficiently find a desired global minimum hidden among many local extrema, a good balance between exploration and exploitation is necessary. Our strategy is to give GAs a richer population and more exploration to avoid unfavorable local minima in early stages. Later, we gradually reduce the number of such minima qualifying for frequent visits. The attention finally shifts more to smaller refinements in the quality of solution.

Instead of constant population size, we introduce a varying population size schedule by utilizing information about the individual fitness. This has also been investigated in [25], [26], [27], [22].

$$p(t+1) = p(t)(1 + \rho) - D(t) \quad (10)$$

where $p(t+1)$ is the population size for next generation; $p(t)$ is the size of current population; ρ is the reproduction ratio and $D(t)$ is the number of expired individuals. The individual lifetime can be determined by

$$\text{MinLT} + \eta \frac{\text{fitness}(C_i^t) - \text{FitMin}}{\text{FitMax} - \text{FitMin}} \quad (11)$$

where FitMax and FitMin are maximal and minimal fitness values found until the current generation, respectively. MinLT is minimal allowable lifetime and η is a constant which can be set to greater than 1. Although varying the population size introduces additional complexity in our implementation, our tests indicate that the output quality improves significantly and the number of generations needed for searching a desired global optimum decreases compared with the canonical genetic algorithm.

The mutation schedule known as non-uniform mutation can be stated as follows [27]:

$$\Delta(t, y) = y(1 - r^{(1-t/T)^b}) \quad (12)$$

where r is a random number in $[0, 1]$, and b is a system parameter determining the degree of nonuniformity, and y

equals to $(q_k - q_k^+)$ or $(q_k - q_k^-)$ for uncertain parameters, $(z - 1)$ or $(z - 0)$ for the absolute value of the root and $(\theta - \pi)$ or $(\theta - 0)$ for the angle of the root. The property of this mutation schedule is to help the GA wander freely among local minima at early stages. The attention of mutation then shifts to smaller refinements in the solution at later stages when we are in the vicinity of a possible global optimum. Based on our tests results, a large mutation rate is suggested. We assume C_i^t and C_{i+1}^t to be two individuals of population $p(t)$ at generation t . Their offspring for the next generation $t + 1$ are generated by linearly combining these two individuals. The resulting offspring are generated by the rules

$$C_i^{t+1} = (1 - \alpha\rho_i)C_i^t + \alpha\rho_i C_{i+1}^t \quad (13)$$

$$C_{i+1}^{t+1} = (1 - \alpha\rho_{i+1})C_{i+1}^t + \alpha\rho_{i+1}C_i^t \quad (14)$$

where α is the value of the random variable distributed on $[0, 1]$. ρ is a *bias factor* that specifies how much additional information is taken from the dominating individual (higher fitness) in the current stage. The new offspring attempt to inherit the better genes from each of their parent. Assuming maximization problems with a non-negative fitness function, ρ can then be determined from the following equations.

$$\rho_i = \begin{cases} \xi_i & : \text{if } \xi_i < 1 \\ 1 & : \text{if } \xi_i \geq 1 \end{cases} \quad (15)$$

$$\xi_i = \frac{\text{fitness}(C_i^t)}{\text{fitness}(C_{i+1}^t)} \quad (16)$$

$$\xi_{i+1} = \frac{\text{fitness}(C_{i+1}^t)}{\text{fitness}(C_i^t)} \quad (17)$$

Implementing these two equations does not raise the genetic algorithm's computational complexity and, based on our numerical tests, the performance of our modified genetic algorithm is significantly better than that of a canonical genetic algorithm. Figure 4 illustrates this crossover operator. When *parent1* shows better fitness than *parent2*, *child1* is allowed to copy additional information from *parent1*. As a result, the chromosome of *child1* retains more similarity with *parent1*.

V. RESULTS

In order to validate our proposed approach, we tested different parametric polynomials with various types of *non-linear dependency*. In all cases, except for a specially constructed example (3) below, the GA successfully tested the sufficiency condition. For brevity, we only consider three of the numerous examples we used to test our approach.

A. Example 1

Consider the following z' polynomial

$$P(z', \mathbf{q}) = a_0 + a_1 z' + a_2 z'^2 + a_3 z'^3 + a_4 z'^4 + a_5 z'^5$$

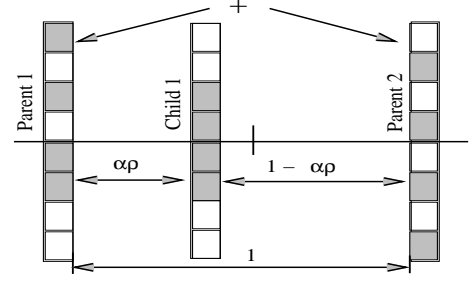


Fig. 4. Our crossover operator

$$a_0 = (q_5^2 - q_1)$$

$$a_1 = -(q_3^2 - 2q_1^2 \sin(\frac{\pi}{2} q_1))$$

$$a_2 = q_2 q_3 q_4$$

$$a_3 = -q_2^2$$

$$a_4 = q_1$$

$$a_5 = q_1 + q_2$$

with $Q = \{\mathbf{q} : q_1 \in [0.5; 1.0], q_2 \in [0.8; 1.5], q_3 \in [0.4; 1.2], q_4 \in [0.5; 1.0], q_5 \in [0.8; 1.7]\}$. The coefficients of polynomial include affine, multilinear, quadratic and trigonometric terms.

The result given in Figure 5 demonstrates the fitness value of an uncertain polynomial found by the GAs as a function of generation. We set the tolerance ϵ equal to 9×10^{-16} as a true numeric zero. The GA stopped when its fitness reaches ϵ . From Figure 5, although the fitness profile barely (visibly) changes after the 40th generation, the GA continuously takes tiny steps to approach a final optimal solution. After running 226 generations, the GA found a pair of roots on the unit circle for $P(z', \mathbf{q})$ at $|1.0|e^{2.84317337937836980j}$ with $\mathbf{q} : [0.998697534807558, 1.492783030649645, 0.424252720951437, 0.982848197210973, 0.808143979297398]^T$ and $P(z', \mathbf{q}) = 5.551E-17 \pm 4.441E-16j$. Since the accuracy is far beyond our requirement, we can safely conclude that there is a root on the unit circle and that the given polynomial $P(z', \mathbf{q})$ is unstable. The computation time required is less than a minute on a PentiumPro-200.

Although detecting a root **on** the unit circle is sufficient to conclude the search, we continued searching for roots **inside** the unit circle. The GA detected a pair of roots at $|0.99448845954136977|e^{0.65541191108445074j}$ after 498 generations. Obviously, searching for roots **on** the unit circle is more efficient than searching **inside** the unit circle. In this example, GAs worked well for testing the sufficient condition of instability for nonlinear uncertain systems with a given accuracy.

It is interesting to visualize how the uncertain parameter is evaluated by the GA. Figure 6 shows changes in the uncertain parameter family during the evolutionary process. The value of parameters in each generation sways severely in an early short period. It then advances forward to the root by making very fine step changes. The GA searches fit structures in the uncertain parameter Q space

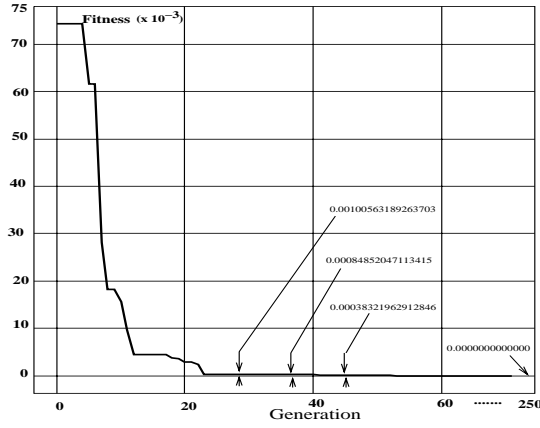


Fig. 5. The fitness of the polynomial $P(z', \mathbf{q})$ as function of generation (Example 1)

and moves toward the global optimum by gradually reducing the chance of reproducing unfit structures, as shown in Figure 6. Unlike gridding techniques, which leads to a combinatoric explosion in the parameter domain, GAs search a much smaller set of structures based on natural selection, and save enough computation time to make a search based approach feasible for this problem.

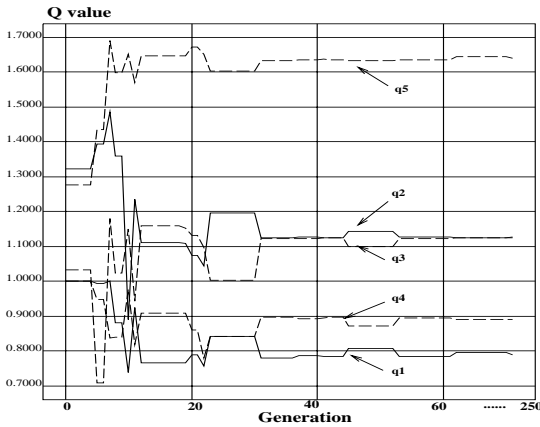


Fig. 6. The uncertain parameter space \mathbf{Q} of $P(z', \mathbf{q})$ as a function of generation

B. Example 2

Consider the following characteristic polynomial

$$\begin{aligned}
 P(z, q, k) &= a_0 + a_1 z + a_2 z^2 + a_3 z^3 \\
 a_0 &= q \\
 a_1 &= \frac{1}{2}[q(1 + e^{-qT}) + k(1 - q - e^{-qT})] \\
 a_2 &= \frac{1}{2}[q(1 + 3e^{-qT}) + k(3 - 2qe^{-qT} - q \\
 &\quad - 3e^{-qT})] \\
 a_3 &= -\frac{1}{2}[qe^{-qT} + k(1 - qe^{-qT} - e^{-qT})]
 \end{aligned}$$

where q is a plant parameter and k is the controller gain. The coefficient functions are exponential for the uncertain

plant parameter q and affine for the controller gain k . Robustness analysis for such an exponential dependence is difficult using conventional techniques. Let the sampling interval T be unity and the feasible region for the uncertain parameters be $\mathbf{Q} \{q, k : q \in [0.1, 2], k \in [0.1, 2]\}$.

We set all parameters of the GA to be the same as in the first example. We map z to z' in the given polynomial by reversing the order of the coefficients then test for the existence of a root *on or inside* the unit circle. Once again, the tolerance ϵ is set to 9×10^{-16} as a true numeric zero. First, we check for roots **on** the unit circle. After 58 generations, the tolerance ϵ converges to 0.00021171124248198 and further improvement is unlikely, as shown in Figure 7. The value of $P(z', q, k)$ is $2.111E-4 \pm 3.6812E-11j$. The accuracy is not satisfactory for our requirement, and therefore further search inside the unit circle is necessary. After running 153 generations searching **inside** the unit circle, the GA finally found a pair of roots at $|0.985708500750795|e^{1.57079632679489700j}$ with $q = 0.10000000000000001, k = 1.427358664247987$, and $P(z', q, k) = 1.6653E-16 \pm 5.4515E-16j$. Once again, using the GA, we can conclude that the given polynomial is unstable.

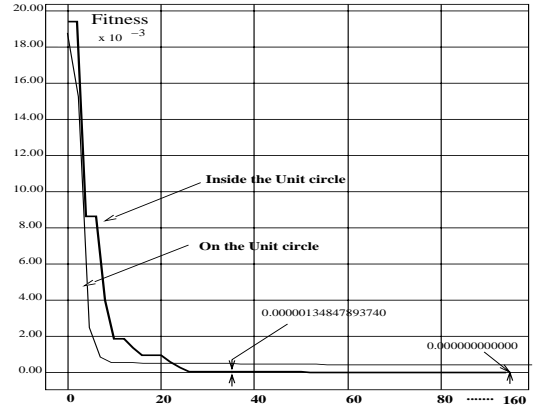


Fig. 7. The fitness of the polynomial $P(z', \mathbf{q})$ as function of generation (Example 2)

C. Example 3

We now consider a simple example that has been specifically constructed to mislead the genetic algorithm approach. Consider a fourth order z' polynomial

$$\begin{aligned}
 P(z', \mathbf{q}) &= a_0 + a_1 z' + a_2 z'^2 + a_3 z'^3 + a_4 z'^4 \\
 a_0 &= q_3^3 - q_2 q_1 \\
 a_1 &= q_1 \sin(\pi q_2) \\
 a_2 &= q_1 q_2^2 q_3 \\
 a_3 &= q_1^2 q_2^2 q_3^3 \\
 a_4 &= -q_1 q_2 q_3
 \end{aligned}$$

The system's uncertain parameter ranges are $Q = \{\mathbf{q} : q_1 \in [1.4; 3.0], q_2 \in [1.0; 2.8], q_3 \in [1.8; 3.2], q_4 \in [1.2; 2.8]\}$. We constructed this polynomial to stress that

our approach only provides sufficiency conditions. A genetic algorithm using the same GA parameters as in the previous two examples was unable to find any roots on or within the unit circle even after 1000 generations. We thus cannot conclude anything about the system's stability from our approach and will need to use some other tool for stability analysis – in this case, an analytic solution is possible.

VI. CONCLUSIONS

In this paper, we demonstrated how genetic algorithms can be applied to aid robust stability analysis. Our approach only provides a sufficient condition for instability for uncertain *nonlinear* parametric polynomials. We modified the GA using novel genetic operators and applied the GA to problems that are difficult to solve using conventional methods due to nonlinearities in the coefficients of their characteristic polynomials. Our computational results indicate that GAs can test the sufficiency condition for system instability with a given accuracy. Furthermore, GAs can be applied to a wider class of systems with large numbers of uncertain parameters. This perspective makes GAs rather unique and promising compared with other approaches.

We also note that if the GA search is unsuccessful we cannot make any conclusions about stability or instability. Our last example illustrates this fact. However, the paucity of tools for dealing with nonlinear parametric polynomials makes our genetic algorithm based approach a viable alternative.

A necessary and sufficient condition for robust stability of discrete-time systems is that all roots of its characteristic polynomial lie inside the unit circle. However, to use genetic algorithms to test this stability condition further investigation is needed. It is our belief that further research in this direction is justified.

VII. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 9624130.

REFERENCES

- [1] E. Jury and T. Pavlidis, "Stability and aperiodicity constraints for systems design," *IEEE Trans. on Circuit Theory*, vol. CT-10, no. 4, pp. 137–141, 1963.
- [2] E. J. Davison, "The robust control of a servomechanism problem for linear time-invariant multivariable system," *IEEE Trans. Automat. Contr.*, vol. AC-21, no. 10, pp. 25–34, 1976.
- [3] J. B. Pearson and Jr. P. W. Staats, "The robust controllers for linear regulations," *IEEE Trans. Automat. Contr.*, vol. AC-19, no. 6, pp. 231–234, 1974.
- [4] R. Barmish, *New Tools for Robustness of Linear System*, MacMillan, New York, 1994.
- [5] V. L. Kharitonov, "On a generalization of a stability criterion," *Izv. Akad. Nauk. Kazakh. SSR Ser. Fiz. Mat.*, vol. 1, pp. 53–57, 1978.
- [6] A. C. Bartlett, C. V. Hollot, and H. Lin, "Root location of an entire polytope of polynomials: It suffices to check the edges," in *Proc. of the 1987 American Control Conference*, 1987, pp. 1611–1616.
- [7] L. A. Zadeh and C. A. Desoer, *Linear Systems Theory*, McGraw Hill, New York, 1963.
- [8] J. Ackermann and B. R. Barmish, "Robust shur stability of a polytope of polynomials," *IEEE Trans. Automat. Contr.*, vol. AC-33, no. 10, pp. 984–986, 1988.
- [9] M. Mansour and F. Kraus, "Strong kharitonov theorem for discrete systems," in *27th IEEE Decision and Control Conference*, 1988, pp. 106–111.
- [10] J. Ackermann, *Robust Control Systems with Uncertain Physical Parameters*, Springer-Verlag, New York, 1993.
- [11] K. A. DeJong, "Genetic algorithms: A 10 year perspective," in *Proc. of the First International Conference on Genetic Algorithms*, 1980, pp. 169–177.
- [12] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
- [13] S. J. Louis, *Genetic Algorithms as a Viable Computational Tool for Design*, Ph.D. thesis, Indiana University, 1993.
- [14] K. Kristinsson and G. A. Dumont, "System identification and control using genetic algorithms," *IEEE Trans on System, man and Cybernetics.*, vol. 22, no. 5, pp. 1033–1046, 1992.
- [15] A. F. Sheta and K. D. Jong, "Parameter estimation of nonlinear system in noisy environments using genetic algorithms," in *Inernational Symposium on Intelligent Control*, 1996, pp. 360–365.
- [16] K. C. Sharman and G. D. McClurkin, "Genetic algorithms for maximum likelihood parameter estimation," in *Proceedings of the IEEE Conference on Acoustic, Speech and Signal Proceeding*, 1989, pp. 2716–2719.
- [17] K. J. Hunt, "Polynomial LQG and H infinite controller synthesis a genetic algorithms solution," *Proceedings of the 31st IEEE Conference on Decision and Control*, vol. 4, no. 865, pp. 3604–3609, 1992.
- [18] D. P. Kwok and F. Sheng, "Genetic algorithm and simulated annealing for optimal robot arm pid control," in *Proceedings of the 1st IEEE Conference on Evolutionary computation*, 1994, pp. 708–713.
- [19] K. S. Tang, K. F. Man, and D. W. Gu, "Structured genetic algorithm for robust H infinite control systems design," *IEEE Trans. on Industrial Electronics*, vol. 43, no. 5, pp. 575–582, 1996.
- [20] Y. Li, K. Chwee, D. J. Murray, and G. J. Gray, "Genetic algorithm automated approach to the design of sliding mode control system," *International Journal of Control*, vol. 63, no. 4, pp. 721–739, 1996.
- [21] C. I. Marrison and R. F. Stengel, "Robust control system design using random search and genetic algorithms," *IEEE Trans on Automatic Control*, vol. 42, no. 6, pp. 835–839, 1997.
- [22] K. Yahiaoui, Y. Haman, and F. Rocaries, "Constrained genetic algorithm-based computer-aided control system design fixed versus variable size population," in *Inernational Symposium on Intelligent Control*, 1997, pp. 425–430.
- [23] A. Varsék, T. Urbančić, and B. Filipič, "Genetic algorithms in controller design and tuning," *IEEE Transaction on System, Man, and Cybernetics*, vol. 23, no. 5, pp. 1330–1339, 1993.
- [24] J. Holland, *Adaptation In Natural and Artificial Systems*, The University of Michigan Press, Ann Arbour, 1975.
- [25] L. Davis, "Adapting operator probabilities in genetic algorithms," in *Proceedings of the 3rd intenational conferece on genetic algorithms*, 1989, pp. 61–69.
- [26] D. Whitley, "The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best," in *Proceedings of the 3rd intenational conferece on genetic algorithms*, 1989, pp. 116–121.
- [27] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs*, Springer-Verlag, New York, 1992.