Now, let us state the main contribution of the paper. The necessary and sufficient conditions in the following theorem hold for *all* center quasi-polynomials $q(z, \mathbf{a}^*)$ $(n > 1)$ and radii r satisfying the Assumption 2.2 in the family $\mathbf{Q}_{\mathbf{w},r}$.

Theorem 4.3: The stability of the eight one-parameter families of quasi-polynomials

$$q(z, \mathbf{a}^*) \pm \left[\lambda \frac{r}{w_{0m}} \pm (1-\lambda)\frac{r}{w_{1m}} z\right] e^{\tau_m z} \tag{29}$$

$$q(z, \mathbf{a}^*) \pm \left[\lambda \frac{r}{w_{(n-1)m}} \pm (1-\lambda)\frac{r}{w_{nm}} z\right] z^{n-1} e^{\tau_m z}, \quad \lambda \in [0, 1]$$

implies the stability of the family $\mathbf{Q}_{\mathbf{w},r}$ if and only if the conditions (26)–(28) in Lemma 4.1 and Lemma 4.2 are satisfied.

Proof: The proof follows from Proposition 2.1, Lemma 4.1, and Lemma 4.2. □

Remark 4.4: Note that similar conditions to (26)–(28) can be stated so that the real weighted diamond of quasi-polynomials enjoys a 12-, a 16-, or more generally, a "weak"-edge property.

Acknowledgment

The authors would like to thank the referees for many helpful comments on the content of this paper. They also would like to thank Dr. A. D. B. Paice for improving the English in the paper.

References

[1] B. R. Barmish, R. Tempo, C. V. Hollot, and H. I. Kang, "An extreme point result for robust stability of a diamond of polynomials," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1460–1462, 1992.

[2] R. Bellman and K. L. Cook, *Differential-Difference Equations*. New York: Academic, 1963.

[3] M. Fu, A. W. Olbrot, and M. P. Polis, "Robust stability for time-delay systems: The edge theorem and graphical tests," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 813–820, 1989.

[4] J. Hocherman, V. L. Kharitonov, J. Kogan, and E. Zeheb, "On the stability of quasi-polynomials with weighted diamond coefficients," *Multidimensional Syst. Signal Processing*, vol. 5, pp. 397–418, 1994.

[5] J. Hocherman, J. Kogan, and E. Zeheb, "Simple stability criterion for quasi-polynomial families with uncertain coefficients and uncertain delays," in *Proc. 32nd IEEE Conf. Decision Contr.*, San Antonio, Texas, Dec. 15–17, 1993.

[6] V. L. Kharitonov and R. Tempo, "On the stability of a weighted diamond of polynomials," *Syst. Contr. Lett.*, vol. 22, pp. 5–7, 1994.

[7] V. L. Kharitonov and A. P. Zhabko, "Stability of convex hull of quasi-polynomials," in *Robustness of Dynamic Systems with Parameter Uncertainties*. Basel: Birkhäuser Verlag, 1992, pp. 63–69.

[8] J. Kogan, "Hurwitz stability of weighted diamond polynomials," *Syst. Contr. Lett.*, vol. 22, pp. 303–312, 1994.

[9] R. Tempo, "A dual result to Kharitonov's theorem," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 195–198, 1990.

Robust Control System Design Using Random Search and Genetic Algorithms

Christopher I. Marrison and Robert F. Stengel

***Abstract*—Random search and genetic algorithms find compensators to minimize stochastic robustness cost functions. Statistical tools are incorporated in the algorithms, allowing intelligent decisions to be based on "noisy" Monte Carlo estimates. The genetic algorithm includes clustering analysis to improve performance and is significantly better than the random search for this application. The algorithm is used to design a compensator for a benchmark problem, producing a control law with excellent stability and performance robustness.**

***Index Terms*—Genetic algorithms, probabilistic methods, robust control design and analysis.**

I. Introduction

Compensators that perform sufficiently well in the presence of plant parameter variations are said to be *robust*. Stochastic robustness analysis is a practical method for quantifying compensator robustness [1]. The stochastic robustness metric characterizes a compensator, $\mathcal{G}$, as the *probability* that the closed-loop system will have unacceptable performance in the presence of possible parameter variations. The probability, P, can be defined as the integral of an indicator function over the space of expected parameter variations

$$P = \int_V I[\mathcal{H}(\mathbf{v}), \mathcal{G}] \mathrm{pr}(\mathbf{v}) d\mathbf{v} \tag{1}$$

where $\mathcal{H}$ is the plant structure, V is the space of possible parameter variations, $\mathbf{v}$ is a point in V, and $\mathrm{pr}(\mathbf{v})$ is the probability density function. $I[\cdot]$ is a binary indicator function that equals one, if $\mathcal{H}(\mathbf{v})$ and $\mathcal{G}$ form an unacceptable system, and is zero otherwise. The designer decides the definition of "unacceptable." For example, it could mean instability, violation of a response envelope, excess use of control, or a combination of qualities. This metric deals equally well with linear, nonlinear, time-invariant, and time-varying systems.

In finding the best compensator, there may be tradeoffs between those that minimize one performance metric and those that minimize a different metric. Tradeoffs can be formalized by combining the probabilities in a scalar robustness cost function

$$J = \mathrm{fcn}(P_1, P_2, \cdots). \tag{2}$$

Each design point, $\mathbf{d}, (\mathbf{d} \in D)$, defines a compensator, $\mathcal{G}(\mathbf{d})$. With I_j, $\mathcal{H}$, V, and $\mathrm{pr}(\mathbf{v})$ fixed, the problem is to find the value of $\mathbf{d}$ that minimizes $J(\mathbf{d})$.

In most applications, (1) cannot be integrated analytically. A practical alternative is to use Monte Carlo Evaluation (MCE) [2] with $\mathrm{pr}(\mathbf{v})$ shaping the random samplings of values for $\mathbf{v}$, and individual selections for each trial denoted by $\mathbf{v}_m$. The trials are repeated for

Manuscript received January 20, 1995; revised January 12, 1996 and July 25, 1996. This work was supported in part by the FAA and NASA under Grant NGL 31-001-252.

C. I. Marrison is with Oliver, Wyman & Company, New York, NY 10103 USA.

R. F. Stengel is with the Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ 08544 USA.

Publisher Item Identifier S 0018-9286(97)03389-8.

N samples in V. The estimates are then

$$\hat{P}_j(\mathbf{d}) = \frac{1}{N}\sum_{m=1}^{N} I_j[\mathcal{H}(\mathbf{v}_m), \mathcal{G}(\mathbf{d})] \tag{3}$$

$$\hat{J}(\mathbf{d}) = \mathrm{fcn}[\hat{P}(\mathbf{d})_1, \hat{P}(\mathbf{d})_2, \cdots]. \tag{4}$$

$\hat{J}$ approaches J in the limit as $N \to \infty$.

In [3], robust linear-quadratic-Gaussian (LQG) compensators for a benchmark control problem were found using MCE and a sequence of line searches. The compensators were exceedingly robust, but the search algorithm was inefficient, requiring many MCE's. This paper develops efficient algorithms for designing robust controllers using random search and genetic algorithms (GA).

II. Random Search and Genetic Algorithms as Used on Deterministic Functions

Although the random search is inefficient, it is simple and often used as a standard for comparison with other searches [4]. The designer initiates the random search by defining the limits of the search space, D. A random number generator selects points $\mathbf{d}_k$ within D, where $k = 1 \cdots N_s$, and N_s is the number of search points. The value of $J(\mathbf{d}_k)$ is tested for each k, and the point giving the lowest value is taken to be the estimate of the global minimizer, $\mathbf{d}^*$.

Genetic algorithms [5]–[7] are randomized adaptive search methods that process a large number of search points at each step and splice the best of the old search points together to produce a new set of points. A GA has two significant advantages for searching a stochastic robustness cost function: randomization within the search method makes the algorithms robust to errors in $\hat{J}$, and the splicing produces implicit parallelism, which limits computational complexity.

The GA manipulates a population of binary vectors representing points in D. There are four operations: evaluation, selection, crossover, and mutation. The initial population is formed by randomly selecting a number (N_{pop}) of vectors $\mathbf{d}_k$ $(k = 1 \cdots N_{\mathrm{pop}})$ from D. $J(\mathbf{d}_k)$ is evaluated and used to select pairs of vectors. The selection procedure is probabilistic and is more likely to choose vectors giving good values of J. Several selection procedures are available [8]; *tournament selection* is used here. Two members of the population are selected, their corresponding values of $J(\mathbf{d}_k)$ are compared, and the member with the better value is retained. A second vector is selected in the same fashion, and crossover is performed on the two retained vectors. A random point is chosen along vectors, separating each vector into a "head" and a "tail," and the two tails are swapped. After crossover, the vectors may be mutated, whereby any element in the vector may be altered with probability P_m. Selection, crossover, and mutation are carried out $N_{\mathrm{pop}}/2$ times to generate a new population. Regeneration is repeated until the population converges.

A real-number genetic algorithm (used here) allows each element of $\mathbf{d}$ to be continuous, but it requires two stages of mutation to affect the local and global search. The first mutation, a multiplicative change, provides the local search

$$d'_{m,k} = d_{m,k} b^{U(-1,1)} \tag{5}$$

where $d_{m,k}$ is the mth element of $\mathbf{d}_k$, b is a base number, typically equal to two, and $U(-1,1)$ is a random variable from a uniform distribution in ± 1. The global search is given by

$$d'_{m,k} = d_{m,k} + U(-1,1)[\max(d_m) - \min(d_m)] \tag{6}$$

where $\max(d_m)$ and $\min(d_m)$ define the limits of direction m of D. Each of these mutations may be given a different probability of occurring.

Elite selection and clustering improve search efficiency. Elite selection retains one or more of the best members of the population and passes them directly into the new generation without crossover or mutation. This ensures that the best solution is not lost. Clustering [9] can create a "super-elite" member to pass into the new generation from groups of vectors that lie close to the global minimum. The centroid is passed to the new generation as a super-elite member.

Values must be chosen for the search parameters within the GA. In the above algorithm these parameters are: the number of population members, N_{pop}, the probability of global mutation, P_{mg}, the probability of local mutation, P_{ml}, the base for local mutation, b, and the initial number of MCE's, N_{MCE0}.

III. Statistical Considerations

If the objective function were evaluated without error, the random search would easily choose the best of the tested points and eliminate all inferior points. However, an MCE allows errors in $\hat{J}$. A search point cannot be eliminated unless there is a significant difference between $\hat{J}$ for the given point and $\hat{J}$ for the best point. For efficient random search, significant difference must be identified using a minimal number of MCE's.

Confidence intervals bound the expected error of an estimate. The probability that the true value of P lies in the interval $[L, U]$ is

$$\Pr[L \le P \le U] = 1 - \alpha. \tag{7}$$

Analytic derivation of confidence intervals is possible only for simple probability distributions. *Bootstrapping* [10] estimates the distribution of $\hat{J}$ by repeatedly simulating the estimation process without reassessing the original function. $\hat{J}'$ is repeatedly estimated as

$$\hat{J}' = \mathrm{fcn}(\hat{P}'_1, \hat{P}'_2, \hat{P}'_3, \cdots) \tag{8}$$

where $\hat{P}'_j$ is sampled from a binomial distribution with a mean of $\hat{P}_j$. The values of $\hat{J}'$ are ordered, the bottom 100α percentile is taken as L, and the $100(1-\alpha)$ percentile defines U.

In searching the stochastic cost function, pairs of compensators, $\mathcal{G}(\mathbf{d}_a)$ and $\mathcal{G}(\mathbf{d}_b)$, are compared on the basis of *estimates* rather than actual costs. If $\Delta\hat{J} \equiv \hat{J}(\mathbf{d}_a) - \hat{J}(\mathbf{d}_b)$ and $\Delta\hat{J}$ is bounded by a confidence interval that does not include zero, there is a statistically significant difference between $\mathcal{G}(\mathbf{d}_a)$ and $\mathcal{G}(\mathbf{d}_b)$. The construction of a confidence interval for $\Delta\hat{J}$ could be based on the confidence intervals for $\hat{J}(\mathbf{d}_a)$, $\hat{J}(\mathbf{d}_b)$ and the Bonferroni inequality, or on the Student t distribution [11]. These methods are conservative in this application because they assume the samples are independent. Tighter confidence bounds on $\Delta\hat{J}$ can be constructed if the same sample points are used for the MCE of both $\hat{J}(\mathbf{d}_a)$ and $\hat{J}(\mathbf{d}_b)$. Although the technique is straightforward, exploiting it is complex; for illustration, consider the simple cost function $J = P_1$. The difference between two compensators is then

$$\begin{aligned}\Delta J &= J_a - J_b = P_{1,a} - P_{1,b} \\ &= \int_V \{I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_a)] - I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_b)]\} d\mathbf{v}.\end{aligned} \tag{9}$$

V has four overlapping subvolumes

$$\begin{aligned} &V_a\colon \mathbf{v} \in V_a \to I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_a)] = 1 \\ &V_{\not a}\colon \mathbf{v} \in V_{\not a} \to I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_a)] = 0 \\ &V_b\colon \mathbf{v} \in V_b \to I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_b)] = 1 \\ &V_{\not b}\colon \mathbf{v} \in V_{\not b} \to I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_b)] = 0. \end{aligned} \tag{10}$$

V_a is the sub-volume of the plant parameter space in which the closed-loop performance with compensator $\mathbf{d}_a$ is unacceptable, and $V_{\not a}$ is the space where it is acceptable. V_b and $V_{\not b}$ are described similarly. Combining these spaces gives (Fig. 1)

$$\begin{aligned} &V_{a \cap b} \equiv V_a \cap V_b, \qquad V_{a \cap \not b} \equiv V_a \cap V_{\not b} \\ &V_{\not a \cap b} \equiv V_{\not a} \cap V_b, \qquad V_{\not a \cap \not b}; \equiv V_{\not a} \cap V_{\not b}. \end{aligned} \tag{11}$$

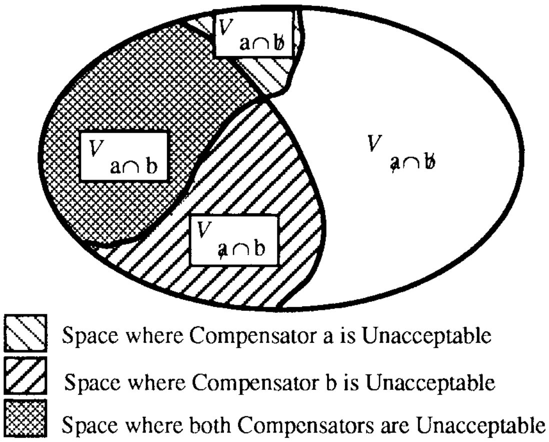

Fig. 1. Division of space according to metrics.

The probabilities in (9) can be restated as

$$P_{1,a} = \int_V I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_a)]\mathrm{pr}(\mathbf{v})d\mathbf{v} = \int_{V_a} I_1[\cdot]\mathrm{pr}(\mathbf{v})d\mathbf{v} + \int_{V_{\not a}} I_1[\cdot]\mathrm{pr}(\mathbf{v})d\mathbf{v}. \tag{12}$$

The indicator function $I_1[\mathcal{H}(\mathbf{v}), \mathcal{G}(\mathbf{d}_a)]$ equals one in V_a and zero in $V_{\not a}$, and (12) becomes

$$P_{1,a} = \int_{V_a} \mathrm{pr}(\mathbf{v})d\mathbf{v} = \int_{V_{a\cap b}} \mathrm{pr}(\mathbf{v})d\mathbf{v} + \int_{V_{a\cap \not b}} \mathrm{pr}(\mathbf{v})d\mathbf{v} = P_{a\cap b} + P_{a\cap \not b}. \tag{13}$$

$P_{a\cap b}$ is the probability of the plant parameter vector, $\mathbf{v}$, being in a subvolume, where both $\mathbf{d}_a$ and $\mathbf{d}_b$ are unacceptable. $P_{a\cap \not b}$ is the probability of being in a subvolume where $\mathbf{d}_a$ is unacceptable and $\mathbf{d}_b$ is acceptable. With these definitions, (9) is

$$\Delta J = (P_{a\cap b} + P_{a\cap \not b}) - (P_{b\cap a} + P_{b\cap \not a}) \tag{14}$$

and it is estimated as

$$\Delta \hat{J} = (\hat{P}_{a\cap b} + \hat{P}_{a\cap \not b}) - (\hat{P}_{b\cap a} + \hat{P}_{b\cap \not a}). \tag{15}$$

If the sampling for each probability is independent, then the variance in the Monte Carlo estimate of $\Delta\hat{J}$ is the sum of variances [11]

$$\sigma^2_{\Delta \hat{J}} = \sigma^2_{\hat{P}_{a\cap b}} + \sigma^2_{\hat{P}_{a\cap \not b}} + \sigma^2_{\hat{P}_{b\cap a}} + \sigma^2_{\hat{P}_{b\cap \not a}}. \tag{16}$$

Alternatively, if the sample points are the same for both $\mathcal{G}(\mathbf{d}_a)$ and $\mathcal{G}(\mathbf{d}_b)$, then $\hat{P}_{a\cap b} \equiv \hat{P}_{b\cap a}$; therefore, (15) and (16) simplify to

$$\Delta \hat{J} = \hat{P}_{a\cap \not b} - \hat{P}_{b\cap \not a} \tag{17}$$

$$\sigma^2_{\Delta \hat{J}} = \sigma^2_{\hat{P}_{a\cap \not b}} + \sigma^2_{\hat{P}_{b\cap \not a}}. \tag{18}$$

Eliminating $\sigma^2_{\hat{P}_{a\cap b}}$ is significant if $\mathbf{d}_a$ and $\mathbf{d}_b$ are close together and have similar performance.

Repeating sample points reduces the estimation error and improves the search. In many cases it is possible to construct tighter confidence intervals around ΔJ, allowing decisions to be based on fewer evaluations. With $J = P_1$, it is a simple matter to construct confidence intervals by bootstrapping. The procedure repeatedly simulates $\Delta \hat{J}'$ as

$$\Delta \hat{J}' = P'_{a\cap \not b} - P'_{\not a\cap b} \tag{19}$$

where $P'_{a\cap \not b}$ is a random number from the binomial distribution with mean $P'_{a\cap \not b}$. For more complex cost functions, ΔJ must be expanded to expose the elements that contain the identical probability estimates, $\hat{P}_{a\cap b}$ and $\hat{P}_{b\cap a}$, which cancel in the construction of confidence intervals. Consider the quadratic cost function $J = \sum_j w_j P_j^2$; then

$$\Delta \hat{J} = \sum_j w_j \left[\hat{P}^2_{j,a} - \hat{P}^2_{j,b}\right] = \sum_j w_j \left[\hat{P}^2_{j,a\cap \not b} - \hat{P}^2_{j,b\cap \not a} + 2\hat{P}_{j,a\cap b}(\hat{P}_{j,a\cap \not b} - \hat{P}_{j,b\cap \not a})\right]. \tag{20}$$

This process eliminates the variation in the random search due to $\hat{P}^2_{j,a\cap b}$ and $\hat{P}_{j,b\cap a}$.

The GA selection procedure is based on differences between cost estimates. Statistical tools are needed to supply sufficiently accurate estimates without using more MCE's than are necessary. Errors in $\hat{J}$ do not affect selection if the error is smaller than the true difference. This principle guides the selection of N, the number of evaluations used to assess each new member. The dispersion of the top 25% of the population is characterized by

$$\sigma^2_{\mathrm{pop}} = \frac{1}{N_{\mathrm{pop}}/4 - 1} \sum_{k=1}^{N_{\mathrm{pop}}/4} [J(\mathbf{d}_k) - \bar{J}]^2 \tag{21}$$

$$\bar{J} = \frac{1}{N_{\mathrm{pop}}/4} \sum_{k=1}^{N_{\mathrm{pop}}/4} \bar{J}(\mathbf{d}_k). \tag{22}$$

The mean variance in the individual estimates of J is

$$\bar{\sigma}^2_{\hat{J}} = \frac{1}{N_{\mathrm{pop}}/4} \sum_{k=1}^{N_{\mathrm{pop}}/4} \bar{\sigma}^2_{\hat{J}(\mathbf{d}_k)} \tag{23}$$

where $\sigma^2_{\hat{J}(\mathbf{d}_k)}$ denotes the variance of estimate $\hat{J}(\mathbf{d}_k)$. $\sigma^2_{\hat{J}(\mathbf{d}_k)}$ is obtained by the bootstrapping procedure [10]. Relating (21) to (23) allows a desirable magnitude of $\bar{\sigma}_{\hat{J}}$ to be defined as $\kappa\sigma_{\mathrm{pop}}$, where κ is a positive number on the order of one, chosen by the designer.

The desired $\bar{\sigma}_{\hat{J}}$ can be related to the required number of MCE's. The stochastic robustness metrics are binary, and $\hat{P}$ has a *binomial probability distribution* [11] with standard deviation

$$\sigma_{\hat{P}} = \sqrt{\frac{P - P^2}{N}} \tag{24}$$

where P is the true probability. The standard deviation of $\hat{J}$ therefore varies inversely with $\sqrt{N}$. If one pairing of N and $\sigma_{\hat{J}}$ is known, then the N required to achieve the desired level of $\bar{\sigma}_{\hat{J}}$ is

$$N_{\mathrm{req}} = N_{\mathrm{cur}}\left(\bar{\sigma}_{\hat{J}_{\mathrm{cur}}}/\bar{\sigma}_{\hat{J}_{\mathrm{des}}}\right)^2. \tag{25}$$

IV. Incorporating Statistical Tools into the Search Algorithms

In the random search, a number of MCE's are carried out for each $\mathbf{d}_k$, using the same sample points to test each compensator. The search estimates the best search point, $\mathbf{d}_{\min}$, as the one with $\hat{J}(\mathbf{d}_k) = \min_j[\hat{J}(\mathbf{d}_j)]$, and for the other compensators calculates $\Delta\hat{J}_k = \hat{J}(\mathbf{d}_k) - \hat{J}(\mathbf{d}_{\min})$. Bootstrapping is used to calculate the lower bound on the confidence interval for $\Delta\hat{J}_k$. This lower bound is $L_{\Delta Jk}$. If $L_{\Delta Jk}$ is greater than zero, then there is a significant difference between $\hat{J}(\mathbf{d}_k)$ and $\hat{J}(\mathbf{d}_{\min})$. When the difference is statistically significant, $\mathbf{d}_k$ can be eliminated from the search, i.e., if $L_{\Delta Jk}$ is greater than zero, $\mathbf{d}_k$ is eliminated.

The reduced population is subjected to additional MCE's. The most efficient number of evaluations is the number that will increase the accuracy of each estimated difference, $\Delta\hat{J}_k$, just enough to eliminate the next test point. This number is estimated by comparing the

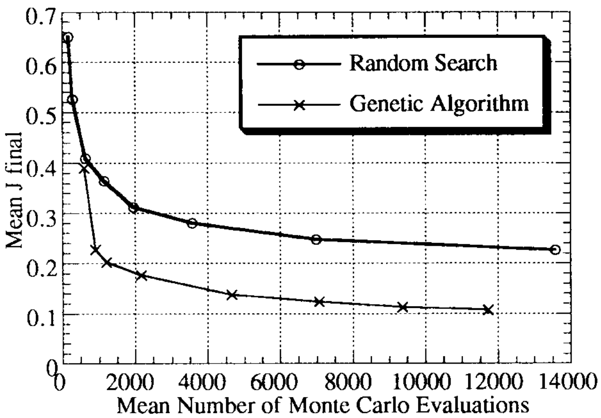

Fig. 2. Performance of the search algorithms on a 24-parameter test function.

current width of the confidence interval around $\Delta\hat{J}_k$ with the interval required to be certain that $\Delta\hat{J}_k$ is greater than zero, i.e., that the lower bound $L_{\Delta Jk}$ is greater than or equal to zero. The current difference between $\Delta\hat{J}_k$ and the lower bound is $(\Delta\hat{J}_k - L_{\Delta Jk})$, and the required difference is equal to $(\Delta\hat{J}_k - 0)$. From (25), the number of MCE's required to eliminate point k is estimated as

$$N_{\text{req},k} = N_{\text{cur}}\left(\frac{\Delta\hat{J}_k - L_{\Delta Jk}}{\Delta\hat{J}_k}\right)^2, \qquad k = 1\cdots N_s. \tag{26}$$

The number of extra evaluations required is $N_{\text{req},k} - N_{\text{cur}}$.

There are two possible stopping conditions for the search. The obvious condition is that only one point remains. The other condition is that the upper bound on all the remaining points is close to the value of the best point, i.e.,

$$\max_k[U_{\Delta Jk}] \le (1+\beta)\min_k[\hat{J}(\mathbf{d}_k)]. \tag{27}$$

$U_{\Delta Jk}$ is the upper bound on the confidence interval for ΔJ_k, and β is a small positive number.

The number of MCE's for each member of the new population (25) must be evaluated for the genetic algorithm. In the process, stratified sampling and retaining a single set of plant parameter vectors within one generation minimizes the variance of $\Delta\hat{J}_k$.

V. Application to a Test Function

The test function maps a point in a 24-dimensional space, D, to a scalar $J(\mathbf{d}),\ \mathbf{d} \in D$. The test function is a weighted quadratic sum of three probabilities

$$J(\mathbf{d}) = P_1^2(\mathbf{d}) + 0.1P_2^2(\mathbf{d}) + 0.1P_3^2(\mathbf{d}). \tag{28}$$

P_1 is the dominant term (e.g., the probability of closed-loop instability), while P_2 and P_3 are less critical performance metrics. Eighteen positive functions are summed to simulate each probability [12]. The minimizing values of $\mathbf{d}$ are in a small portion of the search volume. Here, $J_{\max}$ is 1.2, J_{mean} is 1.071, and $J_{\min}$ is 0.033. Only 3% of D has $J < 0.5$, and 0.09% has $J < 0.25$.

To simulate the effects of MCE, when $P_i(\mathbf{d})$ is sampled, $I_i(\mathbf{d})$ returns the value one or zero. A random value is chosen from a uniform $U(0,1)$ distribution. If the value is less than $P(\mathbf{d}),\ I_i(\mathbf{d})$ returns one; otherwise, zero is returned.

The search performance is defined as the average value of $J(\mathrm{d}_{\min})$, where $\mathbf{d}_{\min}$ is the final selected value. The average value was calculated by running the entire search algorithm 400 times and taking the mean of the best point values. The performance depends on the number of search points chosen (Fig. 2). With 10^4 function evaluations, the random search achieves an average final value $(E[J_{\text{final}}])$ of 0.236 (only 0.055% of D has $J < 0.236$).

The genetic algorithms also were run 400 times against the test function. The minimum value of J was recorded when the search had used more than 2000, 4000, 6000, 8000, and 10^4 MCE's. The baseline algorithm uses an initial set of guesses for the parameters within the GA and does not use clustering or elite selection. $E[J_{\text{final}}] = 0.183$ for a mean number of 10^4 evaluations. Elite selection and clustering improved performance. With 10^4 evaluations, the average performance was $E[J_{\text{final}}] = 0.156$ (0.0035% of D has $J < 0.156$).

Performance was enhanced by adding clustering and elite selection and by tuning the search parameters. The initial and optimized constant values are listed in Table I. After 10^4 evaluations, the random search achieves $E[J_{\text{final}}] = 0.24$, whereas the GA achieves $E[J_{\text{final}}] = 0.11$ (Fig. 2). The random search requires 10^4 evaluations to achieve $E[J_{\text{final}}] = 0.24$, while the GA requires only 900 evaluations to achieve the same result, an 11-fold savings in computation.

TABLE I
Constants of the Genetic Algorithm

Constant	Initial Value	Optimized Value
κ	1	8
N_{MCEo}	8	4
N_{pop}	100	50
P_{mg}	0.01	0.005
P_{ml}	0.1	0.05
b	2	2

VI. Application to a Benchmark Control Problem

The genetic algorithm was used to design robust compensators for a benchmark problem [14]. The plant is a mass-spring-mass system with noncollocated sensor and actuator. The actuator/output transfer function is

$$\mathcal{H}_{uy} = \frac{(k/m_1 m_2)}{s^2[s^2 + k(m_1+m_2)/m_1 m_2]} \tag{29}$$

where m_1 and m_2 are the masses and k is the spring constant, each of unit value. The task is to design a compensator to command u, given measurements of y. There are three requirements: a nominal 15-s settling time in response to a unit disturbance impulse, minimal actuator use, and maximal stability robustness when m_1, m_2, and k are uncertain [15]. The plant parameters have uniform probability distributions, with $0.5 < k < 2, 0.5 < m_1 < 1.5$, and $0.5 < m_2 < 1.5$.

Stochastic robustness analysis was used to compare the robustness of ten compensators designed by five different groups [14], [15]. The robustness was quantified by the probability of instability, (P_i), the probability of excessive settling time, (P_{Ts}), and the probability of excessive actuator use, (P_u). In [3], robust LQG regulators were designed using stochastic robustness metrics, and a line search was used to find parameters that minimize a robustness cost function. Design 1 minimized a cost function defined as $J = P_i^2 + 0.01P_{\text{Ts}}^2 + 0.01P_u^2$, achieving $P_i = 0.0034,\ P_{\text{Ts}} = 0.7588,\ P_u = 0.1077$, or $J = 0.0059$. Design 1 is 1.5 times better than the best of the ten compensators in [15]. However, the synthesis of Design 1 required a total of 701 250 MCE's.

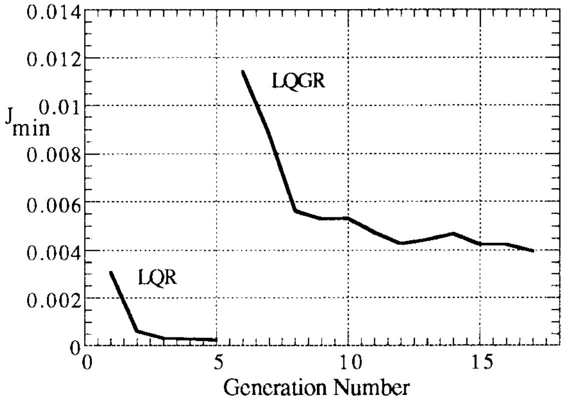

Fig. 3. Progress of the genetic algorithm in finding a compensator for the benchmark problem.

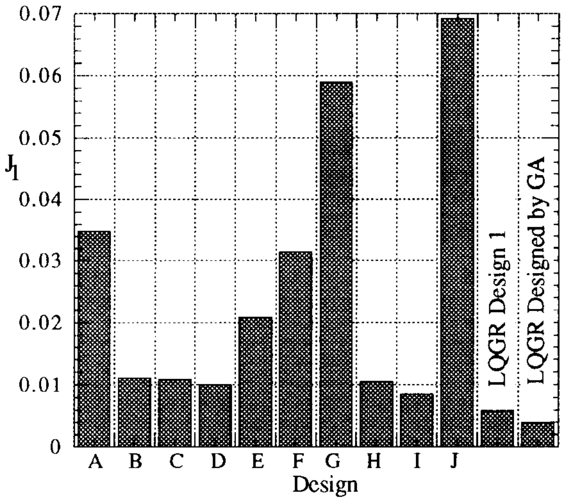

Fig. 4. Comparison of robustness of compensators designed for a benchmark problem (designs A to J correspond with the designations given in [16]).

The GA found a better compensator with one-tenth of the calculations. The same LQG regulator structure was used as in [3], and the same design parameters were chosen. The progress of one run of the genetic algorithm is shown in Fig. 3. Over 20 runs, the search achieved a mean of $\hat{J} = 0.0059$ with a mean number of 70 400 evaluations. After an average of 115 500 evaluations, the mean cost was $J = 0.0056$. The best of the 20 runs achieved $\hat{J} = 0.0039$. For this design $P_i = 0.019$, $P_{\mathrm{Ts}} = 0.596$, and $P_u = 0.019$ (greater stability robustness could easily be obtained by changing the cost function to increase the weight on P_i, but this would sacrifice performance robustness). The resulting compensator transfer function is

$$\mathcal{G}_{yu} = \frac{1.53(s-5.88)(s+0.149)(s+0.479)}{(s+0.693)(s^2+1.54s+5.45)(s^2+2.59s+1.69)}. \quad (30)$$

The nominal closed-loop system has a settling time of 13.2 s and a peak actuator use of 0.52 units. This compensator's robustness compares well with that of the earlier designs, as shown in Fig. 4 [3], [15].

VII. Conclusion

Statistical principles required to efficiently search a stochastic robustness cost function have been developed. The principles are incorporated in random search and genetic algorithms. This work shows that genetic algorithms, combined with statistical tools, provide an efficient method for searching a stochastic robustness cost function, opening new possibilities for the synthesis of robust control systems. The utility of our approach was demonstrated on a benchmark problem, resulting in a design with excellent robustness characteristics.

References

[1] L. R. Ray and R. F. Stengel, "Stochastic robustness of linear-time-invariant control systems," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 82–87, Jan. 1991.
[2] M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods*. New York: Wiley, 1986.
[3] C. I. Marrison and R. F. Stengel, "Stochastic robustness synthesis applied to a benchmark problem," *Int. J. Robust Nonlinear Contr.*, vol. 5, no. 1, pp. 13–31, Jan. 1995.
[4] K. A. DeJong, "Analysis of the behavior of a class of genetic adaptive systems," Ph.D. dissertation, Univ. Michigan, Ann Arbor, 1975.
[5] K. Krishnakumar, "Genetic algorithms: An introduction and an overview of their capabilities," in *Proc. 1992 AIAA GNC Conf.*, Hilton Head, Aug. 1992, pp. 728–738.
[6] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[7] J. M. Fitzpatrick and J. J. Grefenstette, "Genetic algorithms in noisy environments," *Machine Learning*, vol. 3, pp. 101–120, 1988.
[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.
[9] A. A. Törn, "Cluster analysis using seed points and density-determined hyperspheres as an aid to global optimization," *IEEE Trans. Syst. Man. Cybern.*, vol. 7, pp. 610–616, Aug. 1977.
[10] B. Efron, *The Jackknife, the Bootstrap and Other Resampling Plans*. Philadelphia, PA: Soc. Industrial Appl. Math., 1985.
[11] L. Sachs, *Applied Statistics: A Handbook of Techniques*. New York: Springer-Verlag, 1984.
[12] C. I. Marrison, "The design of control laws for uncertain dynamic systems using stochastic robustness metrics," Ph.D. dissertation, MAE-2001-T, Princeton Univ., Princeton, NJ, Jan. 1995.
[13] B. Wie and D. S. Bernstein, "A benchmark problem for robust control design," in *Proc. 1990 Amer. Contr. Conf.*, San Diego, CA, May 1990, pp. 961–962.
[14] ——, "Benchmark problems for robust control design," *J. Guid. Contr. Dyn.*, vol. 15, no. 5, pp. 1057–1059, Sept. 1992.
[15] R. F. Stengel and C. I. Marrison, "Robustness of solutions to a benchmark control problem," *J. Guid. Contr. Dyn*, vol. 15, no. 5, pp. 1060–1067, Sept. 1992.