

AN ITERATIVE CONCURRENT SUBSPACE ROBUST DESIGN FRAMEWORK

Dhanesh Padmanabhan *

Stephen M. Batill †

Department of Aerospace and Mechanical Engineering
University of Notre Dame
Notre Dame, Indiana

Abstract

The purpose of this paper was to evaluate a framework for obtaining robust designs of complex, coupled systems. This framework referred to as Iterative Concurrent Subspace Robust Design (ICSRD), is based upon the use of global response surface approximations of the design space. ICSRD incorporates a robust optimization formulation, using a linearization approach. It generates approximate robust designs from artificial Neural Network (NN) approximations in an iterative fashion. Two benchmark problems are presented, one being an analytic problem with two design variables and the other a control-structures problem, which is characterized by complex discipline coupling. Two variations of the latter problem are considered, one with modified bounds for certain design variables and the other with a reduced number of design variables with original bounds. It is observed that the NN training plays a significant role in obtaining a good robust optimum. It is also observed that ICSRD framework yields reasonable robust designs for the test cases implemented.

Nomenclature

μ_r Mean of Random Variable r
 σ_r Standard Deviation of random variable r

K_r Variance-Covariance Matrix of random vector \mathbf{r}
 $\bar{\mathbf{x}}$ Mean or current Design Variables
 $\bar{\mathbf{y}}$ Mean or current State Variables
 $\bar{\mathbf{p}}$ Mean or current Parameters
 f Merit Function
 g Inequality constraint
 ϵ_r Bias in random variable r
 α Trade-Off parameter
 β Confidence limits for the approximate system
 β_0 Confidence limits for the actual system

Introduction

The ICSRD framework is an extension of the Concurrent Subspace Design Framework (CSD)¹ that was developed for multidisciplinary design of a coupled, non-hierarchical system, with the capability of optimizing systems characterized by mixed discrete, block and continuous design variables and a single performance function. The ICSRD framework incorporates a robust optimization formulation whereas CSD framework does not.

A complex coupled system can be decomposed into Contributing Analyses (CAs) whose outputs are the system's State Variables (SVs). In a coupled system, CAs exchange information with each other, often necessitating the need for iteration to determine the final SVs for a given design point. This process is referred to as the System Analysis (SA). The performance function and constraints are functions of the SVs and the Design Variables (DVs).

*Graduate Research Assistant, Student Member AIAA

†Associate Dean and Professor, Associate Fellow AIAA.

Copyright ©2000 by Stephen M. Batill. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

In the CSD framework, an initial database of DVs and corresponding SVs is created by performing SAs for a selected set of candidate designs. This database is used to develop neural network response surface approximations that replace the “actual” coupled CAs with uncoupled approximate CAs. Optimization is performed on this approximate system model. For systems with computationally expensive SAs, CSD is more efficient than conventional optimization, since SAs in CSD are explicit in the system design variables and design searches in the approximate design space are very efficient. CSD also allows subspace designers to suggest new designs that are used to improve the response surfaces in a cyclic process.

Uncertainty in a CSD framework

The uncertainties in the system design process arise due to uncertainties in design variables, parameters and CAs. Uncertainties in the continuous design variables and parameters can be considered as random effects. Uncertainties in the CAs can be either modeled as biases, if it is known that the CAs always under-estimate or over-estimate the state variables, or as random effects². The use of response surface approximations give rise to additional uncertainty, which is low for high fidelity approximations and vice-versa. Uncertainties are usually assumed to be normally distributed, which is justified by the “Central Limit Theorem” when the uncertainties are due to interaction of many random effects.

In any response surface approximation, it is very difficult to characterize the uncertainty, due to the error of approximation, when the nature of the actual system behavior is unknown. In neural network response surface approximations, this uncertainty can be kept to a minimum by training the neural networks to tolerances that will yield neural networks with “optimal” architecture. Too small a network gives a very poor approximation, and too large a network results in over-fitting and can result in poor generalization.

Robust Optimization

The two main factors taken into consideration in this study are the performance function variation and the probability of failure, due to violation of constraints. Both have to be kept to a minimum. Traditionally robust design is done using Design of Experiments, based on Orthogonal Arrays, due to Taguchi³. One of the typical robust optimization problem formulations, similar to

those used in References 4-6, and based on statistical concepts is as follows:

Formulation 1:

$$\begin{aligned} \text{Minimize } & \mu_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \alpha * \sigma_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) \\ \text{s.t. } & \mu_{g_i}(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \beta * \sigma_{g_i}(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) \leq 0 \\ & \text{for } i=1..J \text{ (no. of constraints)} \\ & \mathbf{x}_l \leq \bar{\mathbf{x}} \leq \mathbf{x}_u \end{aligned}$$

where $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ and $\bar{\mathbf{p}}$ are the current (mean) design variables, state variables and parameters respectively, $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}})$ is the performance (merit) function and $\mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}})$ are the constraints. μ and σ represent the mean and standard deviation respectively. α is a positive weighting parameter that provides a trade-off between obtaining minimum mean and minimum standard deviation of the performance function. It can be set to zero when performance function variation is not an issue. β is a positive parameter that sets confidence levels, of a desired probability, for each constraint about its mean. Greater the value of β , the more conservative the robust design. One could avoid the use of α , if a maximum allowable performance variation is specified by the addition of a constraint in performance function variation. The choice of β is easy when the uncertainty in the g_i s are approximately normally distributed. For non-normally distributed g_i s the choice of β is not straightforward. There are statistical methods that could be used to “fit” known distributions to a Monte-Carlo sample and β could be chosen from the properties of the chosen best “fit” distribution. It should be noted that, one could obtain an upper bound on the value of β for any distribution, for a given probability of failure, from the Chebyshev inequality⁷, but such an upper bound is very conservative and often unnecessary.

The mean and variance can be calculated using either a linearization approximation or a Monte Carlo approach. Linearization is a common way of estimating uncertainties in engineering. Linearization works very well for functions and constraints that are not highly nonlinear. The expressions for the means and standard deviations of the performance function and constraints, using the linearization approach, are as given below.

$$\begin{aligned} \mu_f &= f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \nabla f_y^T \boldsymbol{\varepsilon}_{y,ca} \\ \mu_{g_i} &= g_i(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \nabla g_{iy}^T \boldsymbol{\varepsilon}_{y,ca} \\ \sigma_f &= \sqrt{\nabla f_x^T K_x \nabla f_x + \nabla f_p^T K_p \nabla f_p + \nabla f_y^T K_{y,ca} \nabla f_y} \\ \sigma_{g_i} &= \sqrt{\nabla g_{ix}^T K_x \nabla g_{ix} + \nabla g_{ip}^T K_p \nabla g_{ip} + \nabla g_{iy}^T K_{y,ca} \nabla g_{iy}} \end{aligned}$$

K_x , K_p and $K_{y,ca}$ are the variance-covariance matrices of continuous design variables \mathbf{x} , parameters \mathbf{p} and uncertainty in state variables, due to uncertainty in CAs alone, respectively. $\epsilon_{y,ca}$ are the variations in the state variables due to biases associated with the CAs.

The derivatives with respect to \mathbf{x} can be obtained using Global Sensitivity Equations (GSEs) ⁸ or finite differencing or can be available in analytical forms. The derivatives with respect to \mathbf{y} and \mathbf{p} can be obtained using finite differencing or may be available in analytical forms. Usually uncertainties in the design variables, parameters and CAs are assumed to be independent of each other or uncorrelated, in which case, the variance-covariance matrices become diagonal. The uncertainty in state variables, due to uncertainty in CAs alone, can be represented with either the bias vector or the variance-covariance matrix, or both. These can be calculated using GSEs as developed in Gu et.al ⁹.

Based on the linearization approach one could make a “worst case” formulation for the constraints and hence, an alternative robust optimization formulation can be written as:

Formulation 2:

$$\begin{aligned} \text{Min.} \quad & \mu_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \alpha * \sigma_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) \\ \text{s.t} \quad & g_i(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \sum_{j=1}^N \left| \frac{\partial g_i}{\partial x_j} (\beta * \sigma_{x_j}) \right| + \\ & \sum_{j=1}^P \left| \frac{\partial g_i}{\partial p_j} (\beta * \sigma_{p_j}) \right| + \sum_{j=1}^M \left| \frac{\partial g_i}{\partial y_j} (\epsilon_{y,ca_j}) \right| \leq 0. \\ & \mathbf{x}_l \leq \bar{\mathbf{x}} \leq \mathbf{x}_u \end{aligned}$$

where $(\beta * \sigma_{x_j})$ and $(\beta * \sigma_{p_j})$ are the magnitude of variation in the current design variable x_j and parameter p_j respectively, about their means, with a probability characterized by the value of β . N , P and M are the number of design variables, parameters and state variables respectively.

A Monte Carlo approach, to quantify probabilistic characteristics, is costly but not infeasible when applied to response surface approximations, since the response surface approximations are computationally very inexpensive. A Monte Carlo approach has an obvious advantage over linearization when dealing with highly nonlinear performance functions and constraints. The disadvantage of a Monte Carlo approach is that the robust optimization is not going to be smooth and convergence is not guaranteed.

The problem formulations 1 and 2 treat the variable

bounds as “soft” constraints, that could be violated due to uncertainty. For cases, when this is not true, the variable bounds could be easily modified, to account for such a case, by adding and subtracting $\beta\sigma_x$ to the lower and upper bounds respectively. In practical engineering problem, one could encounter both “soft” and “hard” constraints (one that cannot be violated). For “soft” constraints, one would have to alter the formulations for those specific constraints, by removing the additional weighted standard deviation term, $\beta\sigma_{g_i}$. The ICSR in the present form treats variable bounds as “soft” constraints and the constraints as “hard” constraints.

The problem formulations given above do not include the uncertainty in the response surface approximation. The uncertainty in the response surface approximation can be approximated as a normally distributed random effect with zero mean varying between plus or minus the root mean square (RMS) error of fit, weighted by a factor, for a desired probability and a statistical unbiased estimate (usually the reciprocal of the number of degree of freedoms in the approximation). The disadvantage of this approach is that it would lead to over-conservative designs, especially at designs where the values of the constraints from the approximate CAs are very small when compared to the contribution from the RMS errors, even though they could be sufficiently close to the values of the constraints of the “actual” CAs. In the present study, attempts have been made to obtain acceptable designs, without the inclusion of the uncertainty in response surface approximations.

Two kinds of uncertainties have to be dealt with when performing a robust optimization using response surface approximations, one being the uncertainty (error) in the mean value and the other being the uncertainty (error) in the standard deviation, due to the statistical variations in the design variables and parameters. The uncertainty in standard deviations of the NN approximations at a given design point, is mainly due to errors in the sensitivities, for a general robust optimization formulation and only due to errors in the sensitivities for a robust optimization formulation using a linearization approach. Since no sensitivity information is used in creating the response surface approximations, it is not possible to quantify this error.

The mean and standard deviation uncertainties become very crucial when the constraints of the robust optimization formulation $(\mu_{g_i} + \beta\sigma_{g_i})$, obtained from the approximations, are not conservative, i.e. not lower than the actual constraints of the robust optimization formulation. One would need to obtain a good approximation, at least in the region of interest, to keep this uncertainty to a minimum. Minor discrepancies between

the approximate constraints and actual constraints, on the non-conservative side, could be handled by artificially increasing the value of β for the robust optimization of the approximate SA and determining the robustness of the actual system for a fixed value of β (β_0) though this may not necessarily help when the standard deviation uncertainties are high. So one would need to calculate sensitivities of the actual SA as a verification, to determine if the approximations have to be improved further “locally”, by means of a weighting procedure that will be discussed in the next section.

ICSRD framework

A flowchart of the ICSRD framework is shown in Figure 1. The present form of ICSRD framework is different from the CSD framework in two aspects, the first is that no subspace designers are present in a ICSRD framework and the second is that weighting of designs are not done in CSD. The current ICSRD framework does not accommodate discrete and block design variables nor “biases” associated with the CAs, though the latter can be accounted for easily, since it affects only the robust optimization. The current ICSRD framework incorporates formulations 1 and 2 (formulation 1 alone was used for the test problems) for the robust optimization, using a linearized approach. The basic approach of the ICSRD framework can be summarized as:

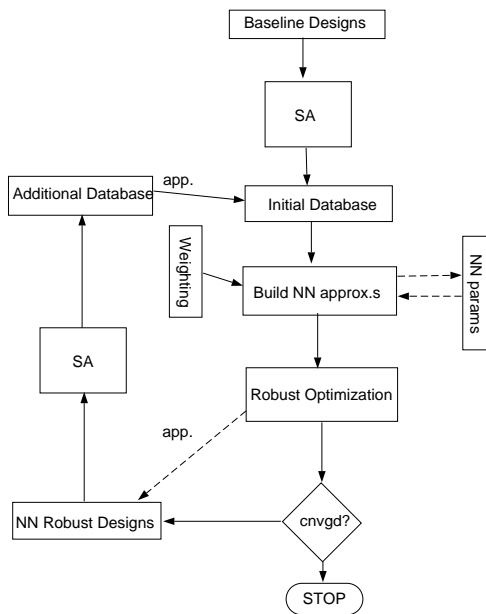


Figure 1. ICSRD (Iterative Concurrent Subspace Robust Design)

1. Generate Initial database (Baseline Designs) and set NN training parameters, optimization parameters, weighting strategy parameters and parameters associated with problem definition.
2. Train NNs and store NN weights and architectures
3. Perform Robust Optimization using NN approximations.
4. Perform SA for the robust design obtained from the previous step and append to the database
5. Is convergence criteria satisfied?, if yes STOP, if no proceed.
6. Determine weights for patterns from a selected weighting criteria.
7. Re-train NNs starting with NN architecture and weights from previous iteration, with the error term modified by the weights determined from previous step
8. Go to step 3.

The ICSRD framework can be divided into the following main stages, that are explained in detail.

Building an Initial Database : An initial database is created from the actual SA, by either a full factorial, fractional factorial or other non-standard approaches. The full factorial approach is very costly when the SA is characterized by large number of design variables and parameters. Some fractional factorial approaches like Central Composite Design (CCD) and Orthogonal Arrays (OA) are less costly. Two possible non-standard approaches are selection of $m_1 \times m_2 \times \dots \times m_n$ uniformly spaced “experiments” or the selection of M random “experiments” within the bounds of the design variables and parameters. The former non-standard approach has been used for one of the test problems, Barnes problem, in the current paper. The latter when used with the Control Augmented Structures problem, did not yield satisfactory results. Hence OAs were used for this problem. The OAs are constructed using a software written by Dr. A. Owen, Department of Statistics, Stanford University. The software can generate Bose, Bose-Bush, Bush and Addelman-Kempthorne designs. One can also use an OA augmented with a few random “experiments”, when more information about the design space is needed. This hybrid approach has not been implemented and is currently under investigation.

Building NN approximations: NN approximations are created using a Conjugate-Gradient based training algorithm¹⁰. The main features of the Conjugate-Gradient training algorithm are the implementation of a bounding algorithm (for bracketing a minimum in weight space),

a golden search routine (for one dimensional minimization), the Polak-Ribiere method and the Powell's restart strategy¹¹. The initial step length for the bounding algorithm and the tolerance for convergence for a golden search routine are user selectable. The other feature is dynamic node creation. A new hidden node is added to the NN when the number of reinitializations of weights exceeds a fixed number. Weights are reinitialized when the rate of decay of the sum of square of errors, for the given training set, becomes less than a given tolerance¹². During training, the best weights and architecture corresponding to the lowest error obtained is stored and updated. When the number of epochs exceeds a preset number (10,000), the weights and architectures are assigned to the best weights and best architecture. The advantage of this training algorithm is that it is faster than a Back Propagation or a gradient descent algorithm. The disadvantage of this training method is that the errors in the patterns need not be uniform and the maximum pattern error can be on the order of the training tolerance.

All the NNs in the ICSR framework have only one output neuron, so the number of NNs trained are either the number of CAs or the number of merit functions and constraints.

Robust Optimization : Robust Optimization routines have been implemented for formulations 1 and 2 using a linearization approach. In the current formulation the robust optimization accounts for uncertainty in parameters and not uncertainty in CAs. Robust Optimization is performed on the NN approximations of the SA but it does not include the error of approximation, which can be quite significant in some cases. The optimization is done for the robust optimization formulation throughout from the starting point, and is not a two-stage robust optimization. Two-stage robust optimization is useful when it is known beforehand that the optimum design obtained by constrained optimization without robust optimization formulation, is nearer to the actual robust optimum. It is assumed that there is no such prior knowledge.

The initial starting point for the optimization for the first global iteration is specified by the user. For subsequent global iterations the previous robust design is used. The disadvantage of this choice of starting point is that the robust designs obtained during the global iterations need not be near to the actual robust optimum and one may end up exploring a region of the design space far away from the "global robust optimum" and may even lead to the failure of the optimization. This is due to the inherent random nature of NN approximations and it also depends on the amount of the information (num-

ber of datasets in the database) and training tolerances to which the NNs are trained. Another strategy would be to keep a fixed initial starting point for all iterations. The disadvantage of this strategy is that, for large problems the optimization could be time-consuming, if the starting point is very far away from the robust optimum. The former strategy has been used in all of the test problems that are presented in the later sections of this paper.

The robust optimization is done using the MATLAB Sequential Quadratic Programming algorithm. The NN training tolerances play an important role in the successful convergence of the robust optimization. The use of very lenient tolerances, can result in a design, which is out of the variable bounds. This problem often occurred for the control augmented structures problem, and this was overcome by training the NNs to stricter tolerances. This problem never happened with the Barnes problem. This is mainly because the control augmented structures problem, has 11 DVs and hence the initial database used to build the NN approximation had less information about the design space than the Barnes problem which is only a 2-DV problem.

Weighting designs obtained from Robust Optimization:

It is desirable to weight designs in the vicinity of the approximate robust design obtained from the previous global iteration, so that the NN approximation obtained by retraining would be a better approximation in the vicinity of previous robust design. This is in tandem with the first strategy of choosing the starting point. These weights are used during NN training to weight the pattern errors corresponding to the designs within the vicinity of the previous robust design.

One can propose various weighting strategies, but one has to be careful not to overtrain the NNs, that would result in larger networks, which are usually bad generalizing approximations. Weighting is done by selecting local designs (L-designs) in the appended database (initial database + actual SAs for the robust designs from previous iterations) within a given user-defined radius from the last robust design (obtained in the previous global iteration). The average NN error, i.e the average of the scaled pattern squared errors, for these L-designs are computed, for each of the NN approximations of the previous iteration. The L-designs are equally weighted with the ratio of this average NN error and a user-defined error tolerance. This error tolerance could be varied during the iterative process, which was done manually for some cases. One could use a higher radius to include the entire appended database, and then set a low tolerance, to improve the approximations. If the approximations are

poor for a set of L-designs alone, one could choose a lower radius and stricter tolerance.

It would be preferable to set lenient tolerances for the errors for L-Designs during the initial stages of the ICSR and stricter tolerances during the final stages of the ICSR. One could also think of not weighting the designs during the initial stages and start weighting sometime in the mid and final stages to enhance the resolution of the NNs, at a desired region in the design space.

Rebuilding NN approximations: The NN approximations are rebuilt at each global iteration. The appended database provides the patterns for rebuilding the NN approximations. Some patterns are weighted, as determined by the weighting strategy. The architecture and the weights of the NNs before retraining are set to those of the NNs of the previous iteration. The NNs for each iteration are trained to the same error tolerance (for the half of the sum of weighted pattern squared errors, summed over all patterns).

Convergence Criteria: To date, manual checks have been done to determine convergence. One could expect the convergence to be obtained when the relative or absolute change in the design point is less than a tolerance level. Such a convergence criteria, does not take into consideration the feasibility and robustness of the actual SA. So a two-stage check for feasibility and robustness is performed in addition to the usual convergence criteria check. At the first stage checks are made to make sure if the mean values of the actual constraints are feasible, if this is infeasible the second stage is unnecessary. At the second stage checks are made to make sure that the actual constraints, as in the robust optimization formulation, are also feasible, i.e. robust. This stage would need an estimate of the actual standard deviations of the constraints, that would need either actual sensitivities or a Monte-Carlo simulation of the actual SA. Only the linearization approach has been taken into account in this paper, so only actual sensitivities are required. When convergence in robust designs are satisfied, but the checks for feasibility fails, one would have to improve the response surfaces by a suitable weighting criteria or by tightening the NN training tolerances. When both the convergence in design point and stage one feasibility are satisfied but the stage two robustness check is not satisfied, and if the difference in the constraints are small, one could artificially increase the value of β and perform another iteration and check robustness for the actual system, for the original value of β , β_0 .

Test Problems

Two test problems, the Barnes Problem¹³ and the Control Augmented Structures¹⁴ problem were chosen as test cases for preliminary assessment of the ICSR framework. Two variations of the latter were considered. For implementation purposes, uncertainties in CAs and parameters were assumed to be absent. Uncertainties were assumed to be present only in the DVs and were taken to be normally distributed. In the test problems considered, the probability of failure of constraints was more crucial than the performance function variation, or in other words, α was set to a low value.

1. Barnes Problem

Barnes problem¹³ is characterized by 2 continuous DV's, 5 SVs, 1 merit function, 21 parameters and 3 constraints. The SA is not coupled. The problem formulation for the Barnes problem is given below.

$$\begin{aligned} \text{Min. } f = & a_1 + a_2x_1 + a_3y_4 + a_4y_4x_1 + a_5y_4^2 \\ & + a_6x_2 + a_7y_1 + a_8x_1y_1 + a_9y_1y_4 \\ & + a_{10}y_2y_4 + a_{11}y_3 + a_{12}x_2y_3 + a_{13}y_3^2 + \frac{a_{14}}{x_2 + 1} \\ & + a_{15}y_3y_4 + a_{16}y_1y_4x_2 \\ & + a_{17}y_1y_3y_4 + a_{18}x_1y_3 + a_{19}y_1y_3 + a_{20}e^{a_{21}y_1} \end{aligned}$$

$$\begin{aligned} \text{s.t. } g_1 = & \frac{y_1}{700} - 1 \geq 0 \\ g_2 = & \frac{x_2}{5} - \frac{y_4}{25^2} \geq 0 \\ g_3 = & (y_5 - 1)^2 - \left(\frac{x_1}{500} - 0.11 \right) \geq 0 \end{aligned}$$

$$\begin{aligned} \text{SVs: } y_1 = & x_1x_2 & y_2 = & y_1x_1 & y_3 = & x_2^2 \\ y_4 = & x_1^2 & y_5 = & \frac{x_2}{50} \end{aligned}$$

$$\text{Bounds: } \quad 0 \leq x_1 \leq 75 \quad 0 \leq x_2 \leq 65$$

Figure 2 shows the contours of performance (merit) function, constraint boundaries, corresponding feasible and infeasible regions, and one of the local optimum, which is the most probable optimum. This problem has more than a single optimum, the global optimum occurs at [75.0, 65.0] where no constraints are active. This global optimum can normally be obtained only if the starting point is near this optimum.

This problem was implemented with the ICSR, by choosing an initial database, that was created by performing actual SAs on an evenly distributed set of 5×4 design points. Four NNs were trained for the merit function and the three constraints. All the NNs were trained to an error tolerance (sum of square of errors with outputs scaled from 0.1 to 0.9) of 0.05. The starting point for the optimization was chosen as [10,10]. α and β were fixed at 2.5. Uncertainties were assumed only for the DVs and were assumed to be uncorrelated and normally distributed with variances 0.1 about the mean of DVs, throughout the entire design space. The robust optimum obtained when performed using the actual SA (not using the NN approximation) with the same starting point was [47.28,18.87] and only the second constraint (of the robust formulation was active).

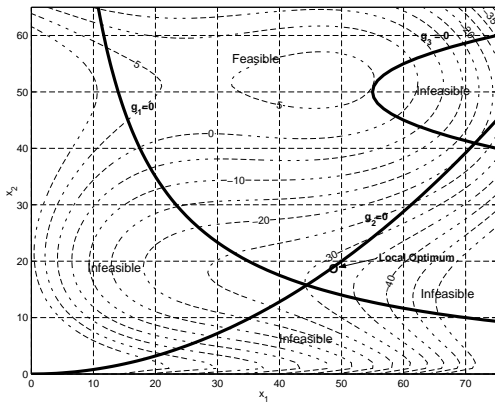


Figure 2. Merit function contours (dashed) and constraint boundaries for actual SA

2. Control Augmented Structures Problem

Control Augmented Structures problem¹⁴ is characterized by 11 continuous DV's (10 beam element dimensions and damping constant), 2 CAs, 1 merit function (total structures and controls weight) and 7 constraints. The SA is coupled. There are 43 SVs. One CA is the controls subsystem and other is the structures subsystem. The aim of this problem is to find the minimum weight of a cantilever rectangular beam, which is divided into 5 elements of equal length and has a controller attached to its free end and is subjected to 3 static loads T_1 , T_2 and T_3 and a dynamic load $f(t)$, as illustrated in Figure 3. The controls subsystem, A & B, is designed as a Linear Quadratic Regulator (LQR) that gives optimal control inputs, to control the beam rotational and lateral displacements. The control effort depends on the eigenfrequencies and eigenvectors of the structures. The controls weight is taken to be proportional to the control effort. The controls weight is in turn required to determine the mass matrix for the structures, which is needed in computation of the static and dynamic properties. The constraints in this problem are due to limitations on the static and dynamic displacements, static and dynamic stresses and natural frequencies for the system. The lower and upper bounds for this problem were initially 3.0 and 36.0 respectively for the first 10 DVs and, 0.01 and 0.06 respectively for the 11th DV.

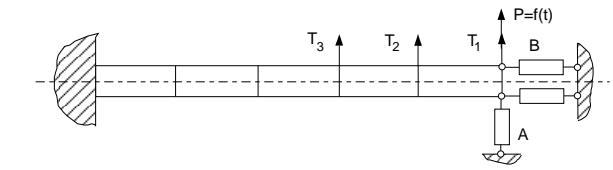


Figure 3. Control Augmented Structures

The actual optimum for this problem, with no uncertainty of any kind, is [3.0, 3.0, 3.0, 3.0, 3.0, 3.7, 7.04, 9.81, 11.99, 13.84, 0.06] with an optimal performance function value of 1493.88 and with the first constraint alone active. It is difficult to find the actual robust optimum, due to the fact that a number of the constraints have discontinuous derivatives which in turn results in jumps in the standard-deviation calculations, at the discontinuities. These discontinuity in derivatives are because the constraints are formulated for maximum static, dynamic displacements and stresses, and these could occur at different elements in the cantilever, at different regions in design space. A Kreisselmier-Steinhauser (KS) formulation could be used to get infinitely continuous smooth cumulative constraints or one could use lenient finite difference parameters to guarantee convergence but this has not been done to date.

Two variations of this problem, as described below, have been considered.

Case 1: An initial database, an OA of 3 levels with 54 points, with first 10 DVs with reduced upper bounds, was chosen. The upper bounds for the first 10 DVs for the actual problem was 36.0, this was reduced to 20.0. NN approximations were built for the controls weight and 7 constraints. All the NNs were uniformly scaled from 0.2 to 0.8. All the training tolerances were set to 0.01 except for constraints 2, 3 and 6, for which training tolerances of $5e-4$, $5e-4$ and $1e-3$ were chosen.

Case 2: The first 5 DVs were fixed at 3.0 and the 11th

DV was fixed at 0.06. An initial database, a modified Central Composite Design for the remaining 5 DVs with original bounds, consisting of 64 points was chosen. The modified Central Composite Design consisted of 2×2^5 points, i.e. corners of two boxes, one that contained the entire design space and the other that shared the same center as the outer box but with its edges half of those of the outer box. As in Case 1, the NN approximations were built for the controls weight and 7 constraints and all the NNs were uniformly scaled from 0.2 to 0.8. The training tolerances were set to 0.01 for the merit function and constraints 3 and 4, $1e-4$ for constraints 1 and 5, and $1e-3$ for the rest.

It can be seen that the initial database for Case 2 has more information about the reduced design space than Case 1, since Case 1 is of higher dimension. For Case 1, the upper bounds were reduced for certain DVs solely to capture more trends in the feasible region with fewer initial designs.

Results and Discussion

Barnes Problem

The convergence histories for the merit function and the worst constraint (for the robust optimization formulation) are shown in Figure 4. The design points and database weighting parameters are shown in Tables 1 and 2. The NN architectures remained 2-4-1 for all the 8 iterations.

Iter No.	x_1	x_2
1	32.31	22.52
2	42.24	18.18
3	43.41	18.40
4	42.49	18.34
5	43.16	18.02
6	43.87	17.91
7	42.97	17.82
8	43.48	17.52

Table 1. Robust Designs generated during ICSR (Barnes Problem)

It can be seen that the NN estimates are very conservative and the robust optimum based upon the NN

Iter No.	Err. tol.	radius	No. of L-Designs
2	$1e-4$	15	2
3	$1e-4$	15	3
4	$1e-4$	15	4
5	$1e-5$	10	3
6	$1e-5$	20	7
7	$1e-5$	20	8
8	$1e-6$	10	6

Table 2. Database weighting parameters and weights (Barnes Problem)

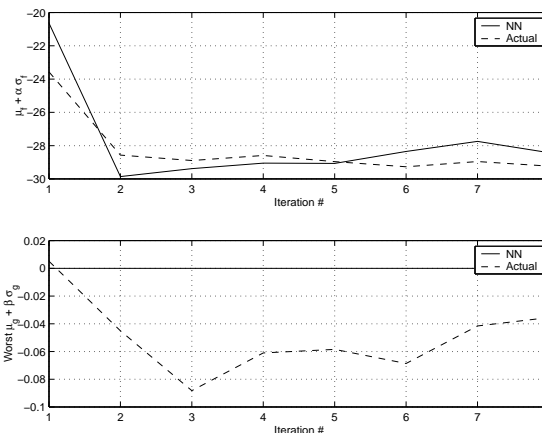


Figure 4. Convergence histories for Barnes Problem

approximations is such that 2 constraints are active, whereas the “actual” robust optimum has only one active constraint. The approximate robust optimum is not very far though, its only slightly below the “actual” robust optimum (obtained from the robust optimization using the actual SA) in the design space. This slight mismatch is mainly due to resolution of the NN approximations. Another notable feature of the convergence is that the robust design oscillates for the last 7 iterations. This is mainly due to the inherent random nature and the resolution of the NN approximations. This was a case where the approximate Robust Optimum was on the conservative side. There is no reason why the approximate Robust Optimum should not be in the infeasible region which was highly characteristic of the 2nd Case of the control augmented structures problem, which is discussed later

in the paper.

Controls Augmented Structures Problem

Case 1: The convergence histories for the merit function and the worst case constraints of the robust optimization formulation are shown in Figure 5. The design points, and the database weights for each NN are shown in Tables 3 and 4 respectively.

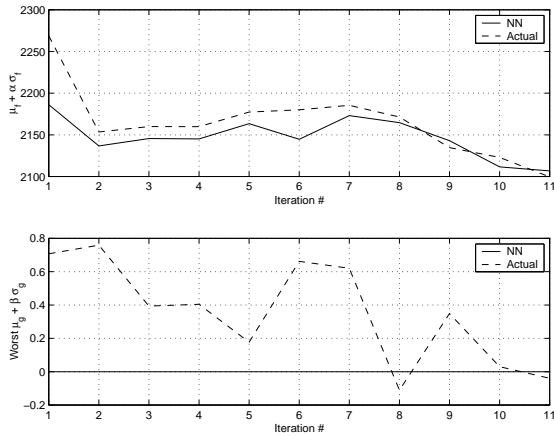


Figure 5. Convergence histories for Structure-Controls Problem, Case 1.

The first five DVs remained at 3.00 for all iterations. At each iteration, the worst case constraint (i.e maximum of $\mu_{g_i} + \beta\sigma_{g_i}$), is primarily due to the 5th constraint. The NN approximation was highly unconservative and the actual SA was infeasible. The training tolerance was made stricter for the 5th constraint, during the 5th iteration. Since this did not improve the approximations, a weighting criteria with strict tolerance, and low radius was chosen to improve the approximation locally in the vicinity of the generated robust designs. It can be seen that as a result, a robust feasible design, for which the mean values of the approximate 5th constraint was conservative as well, was obtained, in iteration 8. Relaxing the weighting criteria worsens the situation, in iteration 9, and making the criteria stringent improved the design in iteration 10 and gave another robust feasible solution in iteration 11.

Case 2: The convergence histories of the merit function and the worst case constraints of the robust optimization formulation are shown in Figure 6. The design points, and the database weights for each NN are shown in Tables 5 and 6 respectively.

Iter No.	x_6	x_7	x_8	x_9	x_{10}	x_{11}
1	3	9.20	14.74	19.59	18.52	0.06
2	6.31	6.03	15.79	13.84	19.88	0.06
3	6.54	6.73	16.39	13.13	19.54	0.06
4	6.71	6.71	16.36	12.85	19.65	0.06
5	7.29	7.61	16.13	12.40	19.61	0.06
6	9.01	7.25	15.90	10.48	20.00	0.06
7	8.73	7.56	16.04	10.61	20.00	0.06
8	10.91	8.42	11.67	14.23	17.85	0.06
9	11.93	6.84	9.73	13.75	19.06	0.059
10	8.79	7.77	11.98	14.14	18.29	0.01
11	8.83	8.03	11.89	14.25	17.86	0.06

Table 3. Robust Designs generated during ICSRD (Structure-Controls Problem, Case 1.)

Iter No.	Err. tol.	radius	No.of L-Designs
2	1e-4	100	54
3	1e-4	100	55
4	1e-4	100	56
5	1e-4	100	57
6	1e-4	100	58
7	1e-4	100	59
8	1e-5	10	5
9	1e-5	20	11
10	1e-6	15	8
11	1e-6	15	9

Table 4. Database weighting parameters and weights (Structure-Controls Problem, Case 1.)

At each iteration, the worst case constraint (i.e maximum of $\mu_{g_i} + \beta\sigma_{g_i}$), is mainly due to the 5th constraint. The NN approximation was unconservative and the actual SA was infeasible, for most of the iterations. After iteration 6, a few combinations of lower radii and stricter tolerances were tried, in an attempt to improve

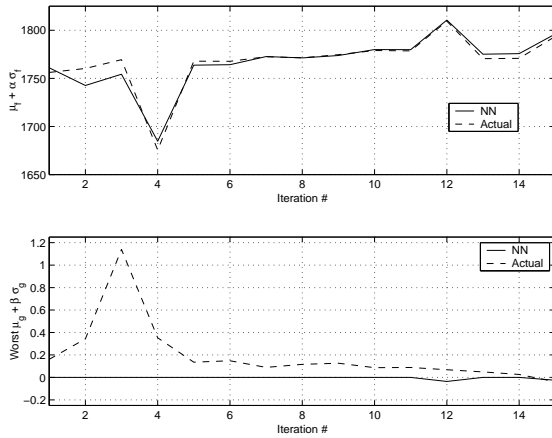


Figure 6. Convergence histories for Structure-Controls Problem, Case 2.

the approximation locally and to get a conservative estimate for the mean values of the constraints. The value of β was increased from 2.5 to 4.0 during iteration 11, but the actual $\mu_{g_i} + \beta_0 \sigma_{g_i}$, with $\beta_0 = 2.5$, was infeasible (> 0), for iterations 11 and 12. So β was restored to 2.5 and weighting criteria was made much stricter, for iteration 13. The mean value of approximate constraint 5 was still unconservative. In iteration 14, the approximate constraint was conservative, in terms of mean value, but the actual $\mu_{g_i} + \beta \sigma_{g_i}$ was infeasible. This showed that the errors in sensitivities could be crucial too. For iteration 15, β was increased to 3.5, and the final NN robust design was feasible, with $\beta_0 = 2.5$.

Conclusions and Future Work

The ICSRD framework was successfully implemented on two test problems and was shown that it was capable of finding acceptable robust designs. Weighting selected designs in the design database was shown to be effective in obtaining good estimates of constraints, by improving the approximation locally. It was observed, as in the control augmented structures problem, that obtaining improved local approximations, that influence certain crucial constraints was important. Hence it was unnecessary to improve approximations that do not influence such crucial constraints and thereby the framework could be made more efficient by implementing a selective weighting criteria by which only those approximations, that influence the violated constraints, are improved locally.

Iter No.	x_6	x_7	x_8	x_9	x_{10}
1	6.95	10.84	11.38	11.70	13.78
2	7.21	10.15	11.69	10.92	14.82
3	7.52	9.58	13.56	13.59	10.05
4	8.60	8.88	9.70	11.72	12.47
5	6.96	10.65	11.07	11.82	14.66
6	6.97	10.62	11.08	11.76	14.76
7	6.52	10.82	11.00	12.06	14.98
8	6.53	10.76	11.10	11.92	15.08
9	6.52	10.80	11.12	11.87	15.15
10	6.47	10.88	11.12	12.07	15.11
11	6.47	10.89	11.12	12.06	15.11
12	6.71	11.10	11.34	12.17	15.46
13	6.38	10.15	10.97	13.84	14.07
14	6.37	10.16	10.92	13.79	14.22
15	6.50	10.23	11.27	13.61	14.67

Table 5. Robust Designs generated during ICSRD (Structure-Controls Problem, Case 2.)

The framework, in its present form, requires design points that require “experiments” with parameters at various levels, to account for the parameter uncertainties. The actual sensitivities due to the parameters are sufficient to account for parameter uncertainties, for a linearization approach. Hence the framework, needs modification to account for this case. Modeling uncertainties and uncertainties due to convergence in a coupled SA have all been neglected, but can be crucial, depending on how the convergence is obtained, during the SA. At each iteration of the ICSRD, only the resulting robust design, is used to improve the approximations. When there is less information about the system, i.e. less points in the database, one may need more designs during each iteration to avoid getting trapped in a local minimum. This is similar to the concept of Subspace Designers in the CSD framework and is currently under investigation. Finally, good NN training is very crucial for the success of an ICSRD framework, and it is very crucial to train NNs to a reasonable strict tolerances to assure the convergence of the robust optimization module.

Iter No.	Err. tol.	radius	No.of L-Designs
2	1e-4	100	64
3	1e-4	100	65
4	1e-4	100	66
5	1e-4	100	67
6	1e-4	100	68
7	1e-6	10	6
8	1e-6	10	7
9	1e-6	15	9
10	5e-7	10	9
11	5e-7	10	10
12	5e-7	10	11
13	1e-7	5	10
14	1e-7	5	12
15	1e-7	5	12

Table 6. Database weighting parameters and weights (Structure-Controls Problem, Case 2.)

Acknowledgments

The authors wish to acknowledge the contributions from many discussions provided by Dr. John E. Renaud, Dr. Amarjit Budhiraja and Ms. Xiaoyu Gu, of the University of Notre Dame. This multidisciplinary research effort was supported in part by the National Science Foundation under grant DMI98-12857.

References

1. Stelmack, M.A., Batill, S.M. and Beck, B.C., "Design of an Aircraft Brake Component Using an Interactive Multidisciplinary Design Optimization Framework", AIAA Paper 99-1346, 40th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Saint Louis, Missouri, April 1999.
2. Batill, S.M., Renaud, J.E. and Gu, X. , "Modeling and Simulation Uncertainty in Multidisciplinary Design Optimization ", AIAA Paper 2000-4803, 8th AIAA/NASA/USAF/ISSMO Multidisciplinary Analysis

and Optimization Conference and Exhibit, Long Beach, California, September 2000.

3. Phadke M.S., Quality Engineering Using Robust Design. Prentice Hall, Englewood Cliffs, NJ, 1989.
4. Sundaresan, S., Ishii, K. and Houser, D.R., "A Robust Optimization procedure with variations on design variables and constraints", Proceedings of 1993 ASME Advances in Design Automation Conference, Albuquerque, NM. Sep 1993, Vol. 65-1, pp. 379-386.
5. Yu, J. and Ishii, K., "A Robust Optimization procedure with variations on design variables and constraints", Proceedings of 1993 ASME Advances in Design Automation Conference, Albuquerque, NM. Sep 1993, Vol. 65-1, pp. 371-378.
6. Su, J. and Renaud, J.E., "Automatic Differentiation in Robust Optimization", AIAA Journal, vol. 35, no.6, June 1997 pp. 1072-1079.
7. Stark, H. and Woods, J.W., Probability, Random Processes and Estimation Theory for Engineers, 2nd Edition, Prentice Hall, Englewood Cliffs, NJ, 1994.
8. Sobieszczanski-Sobieski, J., "Sensitivity of Complex, Internally Coupled Systems", AIAA Journal, vol. 28, no. 1, Jan 1990 pp. 153-160.
9. Gu, X., Renaud, J.E. and Batill, S.M., "An Investigation of Multidisciplinary Design Subject to Uncertainty", AIAA Paper 98-4747, 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Saint Louis, Missouri, Sep. 1998.
10. Battiti, R., "First- and Second-Order Methods for Learning: Between Steepest Descent and Newton's Method", Neural Computation, 4, pp.141-166, 1992.
11. Reklaitis, G.V., Ravindran, A., Ragsdell, K.M., Engineering Optimization : Methods and Applications, John Wiley & Sons, Inc., 1983.
12. Ash, T., "Dynamic Node Creation in Backpropagation Networks", Connection Science, Vol.1, No.4, 1989.
13. Wujek, B.A., "Automation Enhancements in Multidisciplinary Design Optimization", Ph.D. Dissertation, July 1997, Department of Aerospace and Mechanical Engineering, Univ. of Notre Dame, IN, Chapter 7.
14. Sobieszczanski-Sobieski, J., Bloebaum, C.L., Hajela, P., "Sensitivity if Control-Augmented Structure Obtained by a System Decomposition Method. AIAA Journal. Vol.29, No.2, February 1990, pp.264-270.

Errata

for Paper No. AIAA-2000-4841, “An Iterative Concurrent Subspace Robust Design Framework”

1. Page 2, bottom of right column

$$\begin{aligned}\mu_f &= f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \nabla_y f^T \boldsymbol{\varepsilon}_{y,ca} \\ \mu_{g_i} &= g_i(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \nabla_y g_i^T \boldsymbol{\varepsilon}_{y,ca} \\ \boldsymbol{\sigma}_f &= \sqrt{\Delta_x f^T K_x \Delta_x f + \Delta_p f^T K_p \Delta_p f + \nabla_y f^T K_{y,ca} \nabla_y f} \\ \boldsymbol{\sigma}_{g_i} &= \sqrt{\Delta_x g_i^T K_x \Delta_x g_i + \Delta_p g_i^T K_p \Delta_p g_i + \nabla_y g_i^T K_{y,ca} \nabla_y g_i}\end{aligned}$$

where Δ_x and Δ_p are the total differential operators w.r.t. \mathbf{x} and \mathbf{p} respectively, whereas ∇ is a partial differential operator. In general,

$$\begin{aligned}\Delta_x f &= \nabla_x f + \Delta_x y \nabla_y f \\ \text{where } \Delta_x y &= \nabla_x y [GSE]^{-T}\end{aligned}$$

Similar expressions hold good for $\Delta_p f$, $\Delta_x g_i$ and $\Delta_p g_i$.

Note : $-T$ denotes inverse of transpose (or transpose of inverse).

[GSE] is the Global Sensitivity Matrix obtained from the Global Sensitivity Equations. \mathbf{y} can be divided into k components $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k$ which are the outputs of CA_1, CA_2, \dots, CA_k respectively. The [GSE] (which is a function of \mathbf{x} and \mathbf{p}) will be a matrix of dimension $M \times M$, where M is the length of \mathbf{y} , and will consist of various rectangular matrices and identity matrices as shown below

$$[GSE] = \begin{bmatrix} I & -\nabla_{y_2} y_1 & \cdots & -\nabla_{y_k} y_1 \\ -\nabla_{y_1} y_2 & I & \cdots & -\nabla_{y_k} y_2 \\ \cdots & \cdots & \cdots & \cdots \\ -\nabla_{y_1} y_k & -\nabla_{y_2} y_k & \cdots & I \end{bmatrix}$$

2. Page 3, left column, in Formulation 2

$$\begin{aligned}\text{Min. } & \mu_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \alpha \boldsymbol{\sigma}_f(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) \\ \text{s.t. } & \mu_{g_i}(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{p}}) + \sum_{j=1}^N \left| \frac{dg_i}{dx_j}(\boldsymbol{\beta} \boldsymbol{\sigma}_{x_j}) \right| +\end{aligned}$$

$$\sum_{j=1}^P \left| \frac{dg_i}{dp_j}(\boldsymbol{\beta} \boldsymbol{\sigma}_{p_j}) \right| + \sum_{j=1}^M \left| \frac{\partial g_i}{\partial y_j}(\boldsymbol{\beta} \boldsymbol{\sigma}_{y,ca_j}) \right| \leq 0.$$

where $\frac{d(\cdot)}{d(\cdot)}$ are total differentials, which are components of the $\Delta(\cdot)$ defined earlier.