Integrated System-Level Optimization for Concurrent Engineering with Parametric Subsystem Modeling

Todd Schuman* and Olivier L. de Weck†
Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139

and

Jaroslaw Sobieski‡
NASA Langley Research Center, Hampton, Virginia, 23681

The introduction of concurrent design practices to the aerospace industry has greatly increased the productivity of engineers and teams during design sessions as demonstrated by JPL's Team X. Simultaneously, advances in computing power have given rise to a host of potent numerical optimization methods capable of solving complex multidisciplinary optimization problems containing hundreds of variables, constraints, and governing equations. Unfortunately, such methods are tedious to set up and require significant amounts of time and processor power to execute, thus making them unsuitable for rapid concurrent engineering use. This paper proposes a framework for Integration of System-Level Optimization with Concurrent Engineering (ISLOCE). It uses parametric neural-network approximations of the subsystem models. These approximations are then linked to a system-level optimizer that is capable of reaching a solution quickly due to the reduced complexity of the approximations. The integration structure is described in detail and applied to the multiobjective design of a simplified Space Shuttle external fuel tank model. Further, a comparison is made between the new framework and traditional concurrent engineering (without system optimization) through an experimental trial with two groups of engineers. Each method is evaluated in terms of optimizer accuracy, time to solution, and ease of use. The results suggest that system-level optimization, running as a background process during integrated concurrent engineering sessions, is potentially advantageous as long as it is judiciously implemented.

Nomenclature

Variables

A_i	=	Component surface area (m^2)
C	=	Cost ($)
h/R	=	Cone height : radius ratio
κ	=	Material cost-per-unit-mass ($/kg)
L	=	Cylinder length (m)
l	=	seam length (m)
λ	=	Seam cost-per-unit-length ($/m)
M_t	=	Total tank mass (kg)
p_n	=	Nominal tank payload (kg)
ρ	=	Material density (kg/m^3)
R	=	Tank radius (m)
σ	=	Component stress (N/m^2)
t_1	=	Cylinder thickness (m)
t_2	=	Sphere thickness (m)
t_3	=	Cone thickness (m)
$\mathbf{x}$	=	Input design vector
Δv	=	Change in velocity (m/sec)
ζ	=	Vibration factor

Abbreviations

BB	Black Box
BLISS	Bi-Level Integrated System Synthesis
CO	Collaborative Optimization
EFT	External Fuel Tank
GA	Genetic Algorithm
GM	General Motors
ICE	Integrated Concurrent Engineering
ISLOCE	Integrated System-Level Opt. for Conc. Eng.
MATE	Multi-Attribute Trade Space Exploration
MDO	Multidisciplinary Design Optimization
NN	Neural Network
RSM	Response Surface Modeling

* Graduate Teaching Assistant, Dept. of Aeronautics & Astronautics, Room 33-218, AIAA Student Member
† Assistant Professor of Aeronautics & Astronautics and Engineering Systems, Dept. of Aeronautics & Astronautics, Engineering Systems Division, Room 33-410, 77 Massachusetts Avenue, deweck@mit.edu, AIAA Senior Member
‡ Senior Research Scientist, Analytical and Computational Methods Branch, MS240, AIAA Fellow

I. Introduction

A. Motivation

MULTIDISCIPLINARY Design Optimization (MDO) has made significant progress in helping design and optimize complex systems and products over the last two decades[1,2]. There exist, however, a number of obstacles that impede further dissemination of MDO in real product development organizations, particularly at the system level. Chief among these impediments is the apparent incompatibility of automated decomposition-based system optimizers with established Integrated Concurrent Engineering (ICE) practices. In ICE the highest level trades in a system are generally explored and resolved interactively, by human designers, using a variety of linked parametric tools (e.g. Excel spreadsheets). In MDO, on the other hand, a system level optimizer automatically seeks designs that maximize one or more objectives, while satisfying all inequality, equality and side constraints of the problem. Human interaction in traditional MDO is restricted to choosing promising start points or to providing preference weights between objectives. Ultimately, one would like to *concurrently* take advantage of the intuition and creativity of human engineers as well as the speed and impartiality of computer-based system optimizers. The principal problem that must be overcome is that the actions of one must be prevented from inadvertently overriding the actions of the other. The method introduced in this paper, *Integrated System-Level Optimization for Concurrent Engineering* (ISLOCE), solves this problem by letting the human design team and the automated system optimizer operate on different – but linked – representations of the same system. The human designers operate on a high-fidelity parametric representation of the system in the *foreground*, while the system level optimizer explores a lower fidelity approximation of the system in the *background*.

Such a scheme raises a number of questions: How are the fore- and background processes linked together? How is the parametric approximation of the subsystems generated? What is the mathematical formulation of such a framework? Can one show – in practice – that this is more effective than ICE without the augmentation of system optimization? We will provide preliminary answers to these questions after a brief review of the relevant literature.

B. Background of ICE

Integrated concurrent engineering is a collection of practices that attempts to eliminate inefficiencies in conceptual design and streamline communication and information sharing among a design team. Based heavily on methods pioneered by JPL's Team X, concurrent engineering practices have been adopted by major engineering company sectors like Boeing's Integrated Product Teams and GM's Advanced Technology Design Studio. Modern engineering teams that are well versed in these practices see a significant increase in productivity (see Figure 1).

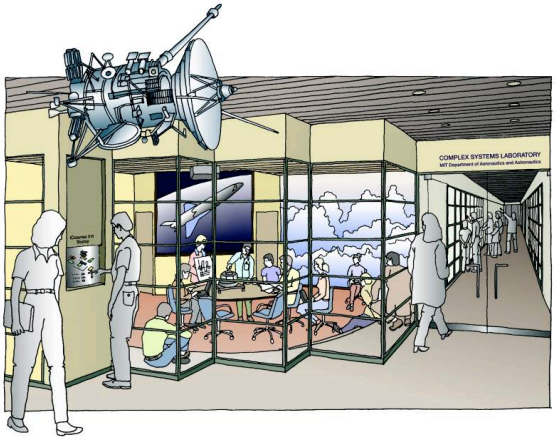


Figure 1. Typical ICE environment

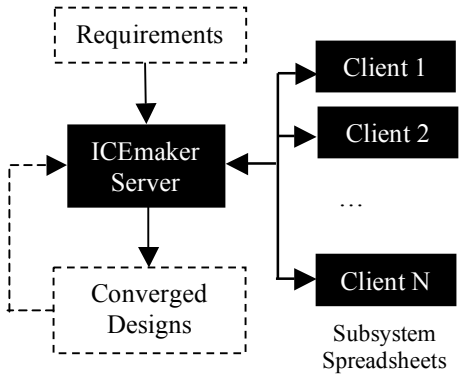


Figure 2. Simplified ICEmaker architecture

Traditional design inhibits interdisciplinary trades, not because they are undesirable, but because of a lack of communication among subsystem teams. Information is scattered throughout the project team, meaning those seeking data on a particular subject have no central location to search. Engineers thus spend a significant fraction of time not developing new information, but rather searching for information that already exists. Integrated concurrent engineering (ICE) has emerged as a solution to the major problems associated with aerospace design, including complicated interdisciplinary interfaces and inefficient time usage. Fundamentally, ICE addresses these issues by:

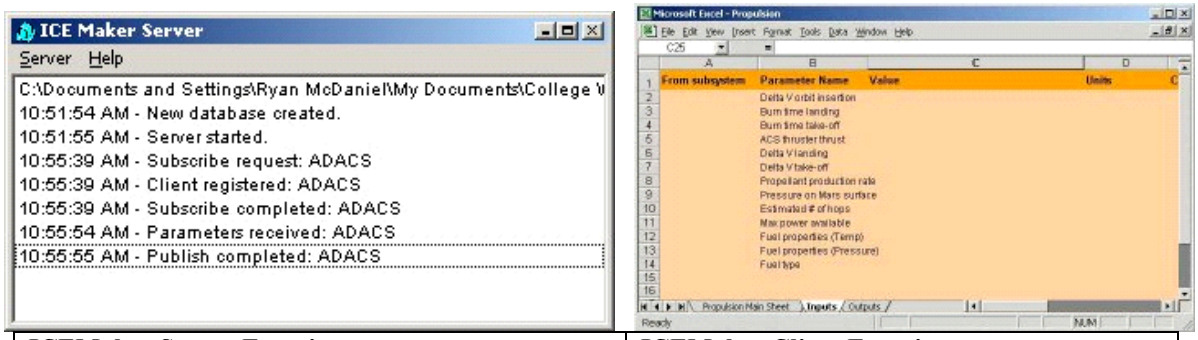
- Encouraging communication between subsystem teams
- Centralizing information storage
- Providing a universal interface for parameter trading
- Stimulating multidisciplinary trades

Once an ICE framework is established, inter- and intra-subsystem trades are clearly defined. This allows teams to work independently on problems local to a subsystem and to coordinate effectively on issues that affect other teams. ICE also provides for near-instantaneous propagation of new requirements. Projects using ICE are more flexible and can quickly adapt to changes in top-level requirements. All these factors together allow engineering teams to conduct rapid trades among complex multidisciplinary subsystems.

C. ICEMaker

Parameter-trading software has become an integral part of ICE teams, allowing users to quickly share information and update each other of changes to the design. This software serves not only as a central server for information storage but also as a tool for establishing a consistent naming convention. Caltech's Laboratory for Spacecraft and Mission Design[3] has made several important contributions to the ICE method under the direction of Dr. Joel Sercel[4]. This includes software known as ICEMaker[5], which was used as a starting point for this research.

ICEMaker is a parameter exchange tool that runs in Excel and facilitates sharing of information amongst the design team, see Fig. 2. ICEMaker has a single-server / multiple-client interface (Fig. 3). With ICEMaker, a design problem is broken down into modules or subsystems with each module ('client') consisting of a series of computer models developed by the corresponding subsystem team. These models are developed offline, a process that can take anywhere from a few days to a few months depending on the desired level of model fidelity. During a design session, each client is interactively controlled by a single team representative ('chair'). The individual clients are linked together via the ICEMaker server (Fig. 2). Chairs can query the server to either send their latest numbers or receive any recent changes made in other clients that affect their work. The querying process is manual, preventing values from being overwritten without permission from the user. Design sessions using ICEMaker typically last three hours and usually address one major trade per design session. Although it has recently become possible to automate this iterative process, human operation of the client stations is almost always preferred. The human element and the ability to catch bugs or nonsensical parameters are crucial to the ICE process. The necessity of keeping humans in the loop will be discussed in greater detail in a later section.



ICEMaker Server Functions:	**ICEMaker Client Functions:**
Maintains parameter list, Adds new clients, Tracks 'send' and 'receive' requests, Error notification	Input, Output and Status sheets for each subsystem, Link to or performs detailed calculations

Figure 3. (left) ICEMaker server sheet and functions, (right) ICEMaker client sheet and functions

During a design session, the server notes all 'send' and 'receive' requests made by the clients. This information is time-stamped so that the facilitator can track the progress of an iteration loop and know how recent a set of data is. In addition to the input, output and status main sheets, a finished ICEMaker client will also have several user-designed sheets. These sheets contain all of the calculations need to process the input data and calculate the output data.

D. Improvements to ICEMaker

While a powerful tool in its own right, attempts have been made to improve upon ICEMaker by incorporating automated convergence and optimization routines into the program. Automatic convergence presents no major problems as the routine simply mimics the role of a human operator by querying each of the clients in turn and updating the server values published by each client. Optimization algorithms have proven more difficult to implement. Each module is usually designed with subsystem-level optimization routines built in that are capable of producing optimal values for the inputs provided to it based on whatever metrics are desired. However, even if every subsystem is optimized in this way, there is no guarantee that the design is optimized at the system level. A system-level optimizer for ICEMaker has been elusive so far, as the human team would be unable to work on a design while the optimizer was running as any values they changed would likely be overwritten by the optimizer as it searched for optimal solutions. Such an optimizer would not be conducive to rapid design and is therefore unsuitable for this problem. It is therefore desirable to develop an optimization method that complements – rather than competes with - the concurrent engineering practices currently in use.

E. Multidisciplinary Optimization

Multidisciplinary optimization is a formal methodology for finding optimum system-level solutions to engineering problems involving many interrelated fields. This area of research has benefited greatly from advances in computing power, and has made possible a proliferation of powerful numerical techniques for attacking engineering problems. Multidisciplinary optimization is ideal for most aerospace design, which traditionally requires a large number of interfaces between complex subsystems. A number of techniques have emerged in an attempt to integrate facilitate both system decomposition and optimization.

One such approach, known as collaborative optimization (CO), has been developed by Kroo, Braun and others at Stanford University[7,8]. This approach divides a problem along disciplinary lines into sub-problems that are optimized according to system-level metrics of performance through a multidisciplinary coordination process. Each sub-problem is optimized so that the difference between the achievable subsystem response and the target variables established by the system analyzer is at a minimum. This combination of system optimization with system analysis is potent, but leads to setups with high dimensionality. This result can drastically increase the amount of processing power needed to run even a simple optimization. CO (like most other distributed methods) is most effective for problems with well-defined disciplinary boundaries, a large number of intra-subsystem parameters and calculations, and a minimum of interdisciplinary coupling. CO has been successfully applied to a number of different engineering problems ranging from vehicle design to bridge construction.

A more recent method, using hierarchical decomposition is bi-level integrated system synthesis (BLISS), developed by J. Sobieski, Agte, and Sandusky at the NASA Langley Research Center[9,10,11]. Like CO, BLISS is a process used to optimize distributed engineering systems developed by specialty groups who work concurrently to solve a design problem. The main difference with CO is that the quantities handed down from the system level in BLISS are not targets, but preference weights that are used for multi-objective optimization at the subsystem level. The subsystems in BLISS are called black boxes (BB) – a designation which we will follow here. Constraints and coupling variables are also handled somewhat differently. The system objectives are used as the optimization objectives within each of the subsystems and at the system level. These two levels of optimization are coupled by the optimum sensitivity derivatives with respect to the design parameters. The design parameters are divided into three groupings. So-called X-variables are optimized at the local level and are found only within each of the subsystems. Y-variables are those which are output by one subsystem for use in another. Finally, system-level design variables are denoted as Z-variables, with system-level denoting variables shared by at least two subsystems.

We will investigate the applicability of CO or BLISS for the background system optimization in future work, but restrict ourselves to a traditional, non-hierarchical system optimization here.

F. Problem Formulation

One of the biggest issues in modern design arises from tension between multidisciplinary optimization and problem decomposition. Decomposing a problem into smaller pieces makes the overall problem more tractable, but it also makes it more difficult for system-level optimization to make a meaningful contribution. Linking together a number of separate (and often geographically distributed) models is not an easy task. As the complexity of the

various subsystems grows, so too does the size of the model needed to perform the system-level optimization. For aerospace designs, an optimization run can take many days or even weeks to finish. *This introduces a factor of lag time into the interaction between the optimization staff and the rest of the design team distances the two groups from each other*. This is a major impediment to full integration of MDO in modern product design. While waiting for the optimization results to come back, the design team presses on with their work, often updating models and reacting to changes in the requirements. When an optimization does finally produce data, the results are often antiquated by these changes. This is of particular concern for concurrent engineering, which strives to conduct rapid design. ICE teams cannot afford to wait for weeks for optimization data when performing a trade analysis. On a more practical level, an integrated engineering team and a computer-based optimizer cannot be allowed to operate on the same design vector, **x**, for fear of overriding each others actions. Thus, we require a framework to mitigate the fundamental conflict between these two approaches throughout the design cycle.

The generic multi-objective system optimization problem formulation is:

Find: $\mathbf{x}^{S*} \in \mathrm{X}$ such that (1)

$$\mathbf{J}^S\left(\mathbf{x}^{S*}\right) \leq \mathbf{J}^S\left(\mathbf{x}^S\right) \quad \forall \quad \mathbf{x}^{S*} \neq \mathbf{x}^S \in \mathrm{X} \quad \text{whereby} \tag{2}$$

$$\mathbf{J}^S\left(\mathbf{x}^S, \mathbf{p}^S, \mathbf{y}^{BB}\right) = \left[J_1^S \quad J_2^S \quad \ldots \quad J_z^S\right]^T \quad \text{is minimized} \tag{3}$$

Subject to: $\mathrm{X} := \mathbf{g}(\mathbf{x}^S) < 0$ and $\mathbf{h}\left(\mathbf{x}^S\right) = 0$ system-level constraints (4)

$$\mathbf{x}_L^S \leq \mathbf{x}^S \leq \mathbf{x}_U^S \quad \text{system-level design vector bounds} \tag{5}$$

$$g\left(\mathbf{x}^{BB}\right) < 0, \quad h\left(\mathbf{x}^{BB}\right) = 0, \quad \mathbf{x}_L^{BB} \leq \mathbf{x}^{BB} \leq \mathbf{x}_U^{BB} \quad \text{and constraints of each BB} \tag{6}$$

Here J_i^S is the i-th System-Level objective and $\mathbf{x}^S, \mathbf{p}^S, \mathbf{y}^{BB}$ are vectors of system design variables, system-level (fixed) parameters and black box sub-system-level responses, respectively.

II. The Integrated System-Level Optimization and Concurrent Engineering (ISLOCE) Method

G. Overview

The ISLOCE method makes use of an optimizer that operates in the background during design sessions without interfering with the work being done by the team members in the foreground (Fig.4).

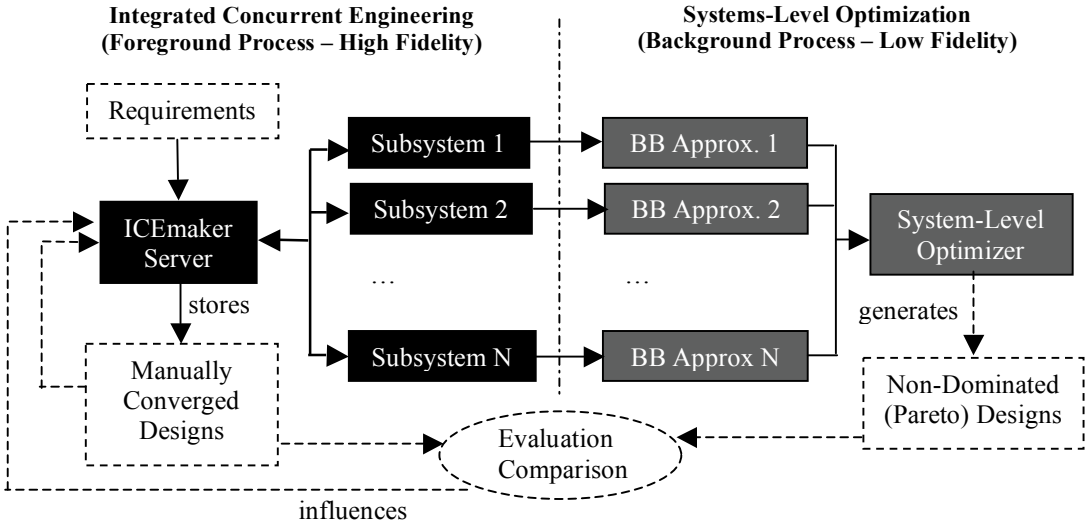


Figure 4. Architecture of ISLOCE framework

The optimizer is initialized before the design session begins and trains a BB approximation[12] (e.g. using neural networks[13]) for each subsystem sheet. This network is not as detailed as the client itself, but instead provides an approximation of its behavior. Once the approximations are constructed, they are saved and effectively act as façades. The optimizer then links them together and runs a heuristic optimization technique (e.g. GA[14], SA) on the system. As the optimizer is working in the background, the human team runs their own design session as normal, periodically referencing the background optimization process for new insights into the problem. As the optimizer begins to identify candidate optimal designs, the ICE session facilitator steers the design team towards those points. It must be reiterated that this approach is not meant to automatically solve problems but is intended to serve as a guide allowing increased session efficiency by quickly eliminating dominated point designs. As the fidelity of the clients grows over time, the human team can export the upgrades to the optimizer and update the subsystem approximations, leading to more accurate designs and a better understanding of the trade space. An illustration of the process is shown in Figure 4.

A major driver for the approach detailed above is that the mating of the two processes be as transparent as possible. ICE and optimization are independently very complex problems already. Any approach that added significantly to this complexity would be useless in a modern design environment. Therefore, an overarching principle of this approach is to integrate optimization with current ICE practices while minimizing the additional work required for either process.

H. Changes implemented to ICEMaker

The decision to use approximations of the ICEMaker modules with the system-level optimizer was based on previous work in distributed processing by Braun, Kroo, Sobieski and Kodiyalam[15]. The current scheme is as follows:

1. An Optimization sheet is added to each subsystem client workbook. This sheet is responsible for generating the neural network approximation of the subsystem model and passing that data on to the system-level optimizer. The only information required from the subsystem chair should be the cell references for the inputs, outputs, and internal parameters of the subsystem. Updating the neural network during a design session should take as little time as possible (on the order of 5-10 minutes).
2. An Optimization subsystem is created and linked to the other clients via the ICEMaker server. The Optimization client is responsible for collecting the neural network data from the other subsystems. Once this data has been assembled, the Optimization client runs a system-level optimizer and generates a set of Pareto-optimal designs using the BB approximations. In a non-hierarchical scheme the execution sequence of BBs and convergence must be automated.

It should be noted that the Optimization subsystem is not a ‘dummy’ client and requires a skilled human chair just like every other subsystem. The operator must be capable of interpreting the optimizer results and passing that information, along with his or her recommendations, to the session facilitator. This implementation method should minimize the impact of integrating optimization with concurrent engineering.

I. Neural Network Approximations

Initial work on the ISLOCE method focused on the code needed to generate the approximations used in the background process. Two candidate solutions were response surface mapping (RSM)[15] and neural networks[13]. RSMs are easier to code and could be implemented directly in Excel by using Visual Basic macros. Unfortunately, they are only well suited to approximate multiple-input/single-output functions. Given that the majority of aerospace project clients have multiple outputs to other subsystems, this would require greatly simplifying the models used in the clients. Neural networks are much more versatile and can approximate functions with large numbers of both inputs and outputs. A neural network consists of layers of neurons, each containing a transfer function and an associated weight and bias, see Figure 5. The network is “trained” by presenting it with a series of inputs and corresponding outputs. The network attempts to simulate the results presented to it by altering the weights associated with the various neurons using least-squares or another minimization routine. If properly trained, a neural network can accurately approximate most families of functions. However, they are much more code intensive and cannot be easily implemented using Excel alone.

For these reasons, the Matlab neural network toolbox is used to construct the NNs used in this project[13]. However, this solution requires a way of generating a large amount of training data from the ICEMaker clients in Excel and feeding it to Matlab. Exporting the data from Excel and then importing to Matlab is possible, but cumbersome. It does not fit with the principle of minimizing additional work for the subsystem chairs. The enabling piece of software that solved all of these difficulties and more is appropriately named “Excel Link”. This add-in for

Excel allows seamless transfer of matrices between Excel and Matlab. Excel-Link coupled with Visual Basic macros allows the generation of data tables and neural networks for any subsystem clients.

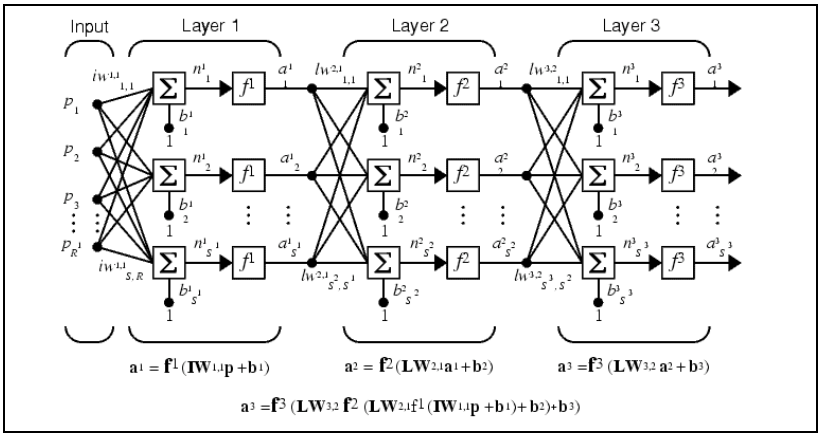


Figure 6. Sample Neural Network with three layers

The neural network generation scheme was incorporated into the external fuel tank model discussed in Section III, to be used in the live trial exercise of Section IV.

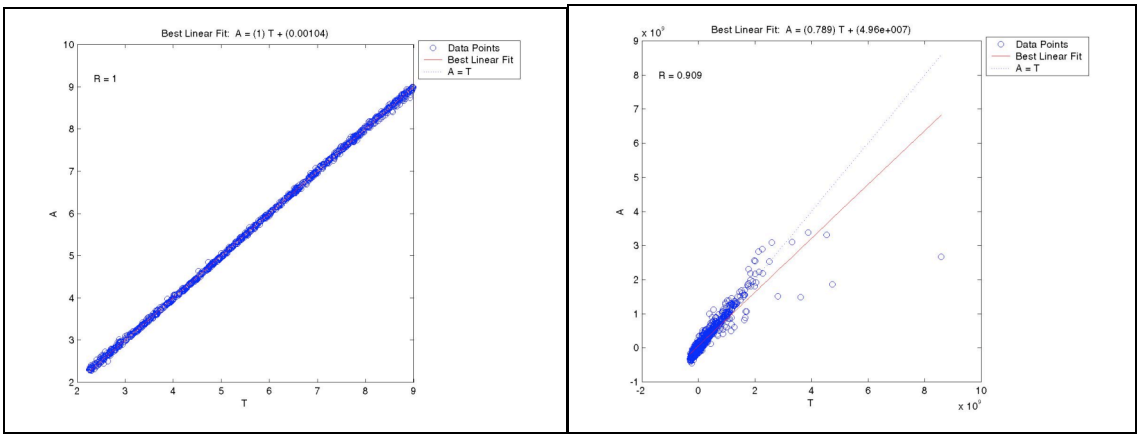


Figure7. (left) Neural network prediction for Structures BB subsystem of EFT. Total tank mass R~1, (right) Neural network prediction for Structures BB subsystem of EFT, Cone stress R~0.91

Figure 7 (left) plots the specific performance of the structures neural network at matching the target value for the total tank mass output. The network performs extremely well at this task, with a regression factor of nearly 1. Most of the outputs for all of the networks were very near this level. However, there were a few outputs that consistently proved problematic to approximate. Figure 7 (right) shows the structures neural network performance at predicting cone stress. The regression factor for this output is only about 0.91. This lower level of performance can be potentially improved through modification of neural network parameters. On average, the neural networks were able to match the values generated by the full-fidelity external fuel tank model extremely well, this is true especially of the cost and aerodynamics subsystems. Some problems generated by mismatches between full fidelity subsystems and NN approximations were discovered during the live trials (Section IV), particularly with respect to constraint violations.

J. Genetic Algorithm Optimizer

The genetic algorithm operated by the Optimization chair during design sessions is based on a third-party GA toolbox for MATLAB. A fitness function was developed that penalized individuals both for being dominated and for violating constraints. The genetic algorithm code was modified slightly to allow for this type of fitness function Once the GA code was developed, the neural networks generated in the previous section were collected and linked

to the optimizer. A number of test runs were performed to verify proper GA behavior. Some sample data from the trial runs is provided in the figures below.

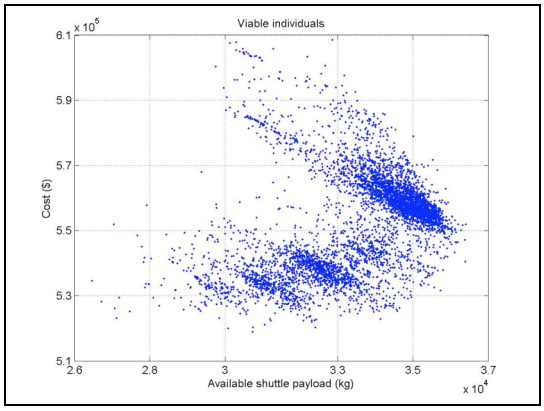


Fig. 8 Converged GA population for EFT problem

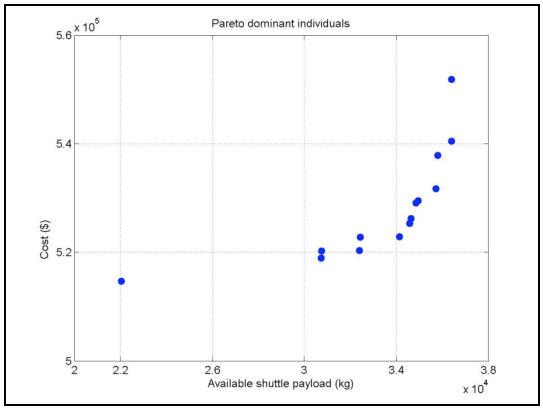


Fig.9 Non-dominated designs extracted from GA

Figure 8 plots the payload-versus-cost performance of all viable (no constraint violation) individuals discovered during the GA run of the EFT sample problem (Section IV). A relatively clear Pareto front develops towards the lower right corner of the plot. Interestingly, the trade space for viable designs does not appear to be evenly distributed, as several thick bands of points can be discerned in the plot. Figure 9 plots the non-dominated individuals from the previous chart on a separate graph. The Pareto front is fairly obvious towards the high-payload region of the curve, but there is a large gap between the "knee" of the curve and the low-payload region. These gaps were frequently found during various GA runs and made it difficult to completely fill in the Pareto front for the EFT trade space. The incorporation of restricted mating and other techniques into the GA code could help spread the Pareto front out along more uniformly (future work).

III. Case Study: STS External Fuel Tank

K. Model Description

The model used in this study is a simplified version of the Space Shuttle external fuel tank (Fig.10) provided by Dr. Jaroslaw Sobieski. It was originally developed as an illustrative tool to demonstrate how changes in a problem's objective function influence the optimal design[16]. This choice of problem was made for several reasons:

- The model is based in Excel (as is ICEMaker), allowing easy integration into the desired test environment.
- The original model could be solved numerically for a variety of objectives using Excel's Solver routine.
- There is sufficient complexity in the model to provide a reasonable test of the method's capabilities
- Many of the participants were already familiar with the model from previous use.

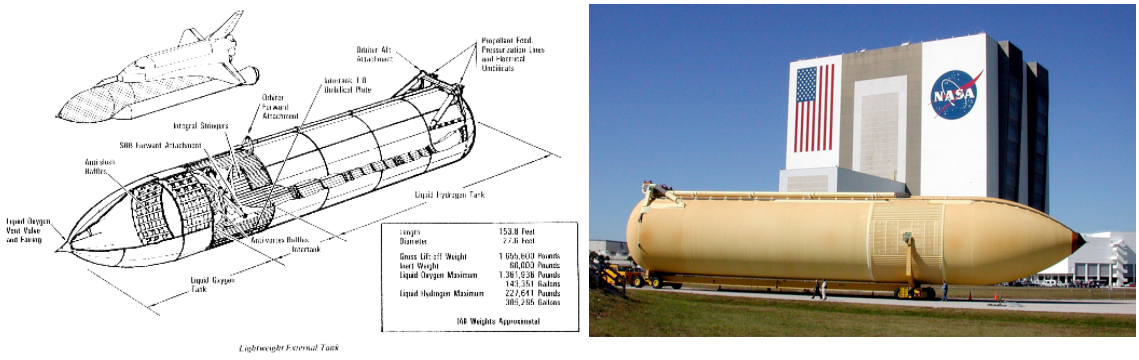


Figure 10. (left) Space Transportation System (STS) External Fuel Tank configuration, (right) External Fuel Tank (EFT) in front of Vehicle Assembly Building at the NASA Kennedy Spaceflight Center

The model divides the tank into three hollow geometric segments: a cylinder (length L, radius R), a hemispherical end cap (radius *R*), and a conical nose (height *h*, radius *R*), see Figure 11. These segments have thicknesses t_1, t_2, and t_3, respectively. Each segment is assumed to be a monococque shell constructed from aluminum and welded together from four separate pieces of material. This results in a total of fourteen seams (four seams per segment times three segments plus the seams at the cone/cylinder and cylinder/sphere interfaces). Surface areas and volumes are determined using geometric relations, and first principles and rules of thumb are used to calculate stresses, vibration modes, aerodynamic drag, and cost. The subsystems are briefly described.

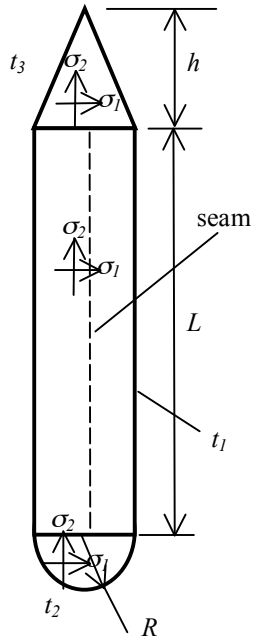


Fig. 11 EFT Model

Structures

Input: six tank dimensions (*L, R, t_1, t_2, t_3, h/R*)

Output: component and tank surface areas and volumes, component and tank masses, stresses, first vibration mode frequency

The volume of the tank is held constant to accommodate an equal amount of propellant regardless of the tank design and serves as an equality constraint. The mass of each component m_i is calculated as:

$$m_i = A_i t_i \rho \tag{7}$$

where ρ is the density of the material used for the tank (Al). Stresses σ_i are calculated based on the assumed internal pressure of the tank and are measured in two directions per component as shown in Figure 11. These calculations result in a component equivalent stress given by

$$\sigma_e = \sqrt{\sigma_1^2 + \sigma_2^2 - \sigma_1 \sigma_2} \tag{8}$$

This equivalent stress may not exceed the maximum allowable stress parameter set within the model. Together, the three component equivalent stresses serve as additional model constraints. A final constraint is placed on the first bending moment of the tank. A vibration constraint ζ is calculated which is proportional to the tank radius and cylinder thickness and inversely proportional to the mass.

Aerodynamics

Input: tank radius *R* and cone height *h* , surface and cross-sectional areas

Output: maximum shuttle payload, m_p

The aerodynamics module computes the resulting drag on the tank during flight. Cone drag is calculated based on empirical trends according to

$$D = b + a \cdot \exp\left[1 - c\frac{h}{R}\right] \tag{9}$$

where *a, b*, and *c* are experimentally determined constants. The drag and surface areas are then compared to nominal values for the original tank. The change in available payload is calculated from a weighted linear interpolation of these comparisons, by

$$m_p = p_n - \Delta M_t + \Delta p \tag{10}$$

where p_n is the nominal payload, ΔMt is the deviation in tank mass from the nominal value, and Δp is the change in available payload described above.

Cost

Input: tank dimensions and component masses

Output: seam (= welding labor) and material costs

The cost module uses the tank dimensions set by the structures module to calculate the seam lengths required to weld each component. A seam's cost is dependent upon its length and the thickness of the material being welded. A base cost-per-unit-length parameter λ is set within the model and is multiplied by the seam length l and an empirical function of the material thickness

$$C_{seam} = l\lambda f(t) \tag{11}$$

with the function *f* given by

$$f(t) = a + b(t - \Delta) + c(t - \Delta)^2 \tag{12}$$

Here, t is the material thickness, Δ is the weld offset, and a, b, and c are industry-determined constants. For the twelve intra-component welds, the thickness t is just the component thickness. The two inter-component welds use the average thickness of the two components in the function $f(t)$. The procedure for calculating material costs is similar. A base cost-per-unit-mass parameter κ is set within the model. This parameter κ is then multiplied by the component mass and another function of thickness. The material cost of all components plus the sum of the seam costs calculated above yields the total cost of the tank.

Systems

Input: tank dimensions, total cost, available shuttle payload

Output: visual representation of tank, running history of tank designs, Pareto front construction

The systems module presents a high-level summary of the overall tank design. It does not perform any direct calculations but instead helps the team to visualize the current tank and track the team's progress as it explores the trade space.

Optimization (optional)

Input: neural network data from the three main modules (structures, aerodynamics, cost)

Output: prediction of expected Pareto front and table of possible Pareto-optimal designs in terms of both their performance (payload) and cost as well as their associated design vectors.

The optimization module is not developed from the original EFT model but is instead an optional add-on to the rest of the system. Based on the ISLOCE method, the optimization module compiles the neural network approximations of the other subsystems and uses this information to run a system-level optimization of the EFT design. The objective vector is chosen to be $\mathbf{J} = \{m_p, c\}$. Eleven constraints are levied on the design space. The first six constraints are a set of limits placed on the input vector (side constraints). These constraints limit modification of the input variables to a range of values that could be realistically developed in industry. The next restriction is an equality constraint on the tank volume (~3000 m^3 +/- 100 m^3). This constraint creates an interesting dilemma in that it is difficult for both humans and heuristic optimization techniques to match such a constraint. In this case, the tank volume is dependent upon three parameters (L, R, h/R) meaning that any two parameters can be free while the third is dependent upon the others. However, no restriction is placed on which parameter is chosen as dependent. Finally, inequality constraints are placed on the maximum allowable component stress and on the first bending moment of the tank. The equivalent stress experienced by each component cannot exceed the maximum allowable stress of whatever material is used. Also, the first bending moment of the tank must be kept away from the vibrational frequencies experienced during launch.

Maximize: $J = \begin{bmatrix} m_p & c \end{bmatrix}^T$ such that $J(\mathbf{x}^*) \geq J(\mathbf{x}) \quad \forall \quad \mathbf{x}^* = \mathbf{x}$ subject to (13)

Design Vector bounds:

$$1 \leq L \leq 100 \Rightarrow |L - 50.5| - 49.5 \leq 0$$
$$2.25 \leq R \leq 9 \Rightarrow |R - 5.625| - 3.375 \leq 0$$
$$1.75 \cdot 10^{-3} \leq t_i \leq 14 \cdot 10^{-3} \Rightarrow |t_i - 7.875 \cdot 10^{-3}| - 6.125 \cdot 10^{-3} \leq 0 \quad (14)$$
$$0.1 \leq \frac{h}{R} \leq 5 \Rightarrow \left|\frac{h}{R} - 2.55\right| - 2.45 \leq 0$$

Volume constraint:

$$2826 \leq V_t \leq 3026 \Rightarrow |V_t - 100| - 2926 \leq 0 \quad (15)$$

Stress and vibration constraints

$$\sigma_{e,i} \leq 4 \cdot 10^8 \Rightarrow \sigma_{e,i} - 4 \cdot 10^8 \leq 0$$
$$0.8 \leq \zeta \Rightarrow 0.8 - \zeta \leq 0 \quad (16)$$

IV. ISLOCE Live Trials (Experiments)

L. Trial Motivation

As mentioned previously, other researchers have developed conceptual design methods similar to the one outlined in this paper. While they all differ in their formulation, each has a common goal: to increase the productivity and efficiency of engineering teams. As such, any new method proposal would be deficient if it did not provide evidence of being able to achieve that goal. It is common practice to demonstrate a process in the context of an appropriate test case. This not only serves as a proof-of-concept but also provides a convenient illustration of the method in action.

Satisfactory demonstration of a new method typically involves successful application of the method to a test case as described above. However, there is no fixed criterion for what is considered a 'success'. Previous papers have usually chosen to apply their methods to a problem for which an optimal solution is already known. The method is then shown to converge to the optimal solution in a reasonable amount of time. If the main purpose of the test is to confirm that the method can reach a solution, then this type of experiment is adequate. However, it neglects several key factors that are of great importance to the engineering teams that will actually use the method for industry-related design. Engineering teams are concerned not just with reaching a single optimal solution, but also completely exploring a trade space and arriving at a family of optimum solutions that covers the range of multiple objectives. Further, time-to-convergence is not as important as ease of setup and use for the design team. A method that is cumbersome to setup, use, and modify does not increase the productivity of the team that uses it. These are all highly relevant issues that are not addressed by a simple convergence test. The only way to evaluate a method accurately according to these metrics is with a live test by actual engineers in a distributed design session.

M. Trial Objectives

To do this, it is necessary to both evaluate the method itself by comparing the results to more conventional design techniques. The well-established principles of the scientific method are applied here by introducing two experimental groups. First, a *control group* uses conventional concurrent engineering practices (with ICEMaker) and the model described above to investigate the EFT trade space. The result should be a family of designs that provides a baseline level of accuracy and group productivity. Then, a *test group* investigates the same problem (with no knowledge of the control group's results) by using the full ISLOCE method, i.e. ICEMaker with system optimization running in the background. Presumably, the two groups will arrive at different sets of solutions that can be compared to each other for accuracy and completeness in terms of defining the trade space and Pareto front. *The hypothesis to be tested is that the use of background optimization will make the test group more effective relative to the control group, given the same amount of time*. The design trajectories taken by the groups can also be compared and ranked according to the number of designs developed, the ratio of dominated/non-dominated designs, elapsed time, etc. After the experiment is run, this information can be collated and analyzed to gain insight into the effectiveness of the ISLOCE method and the role of optimization in conceptual design.

Trial Objectives: The task presented to both groups is identical. Given the EFT model, each group attempts to solve the multidisciplinary design optimization problem posed above (Eq.13-16) within a fixed amount of time. The end result should be an approximation of the EFT Pareto front with designs that maximize available shuttle payload, m_p, minimize tank construction costs, c, and satisfy all constraints on volume, stress, and vibration. The secondary goals are those that allow additional insight into the effectiveness of the method used. The ratio of the number of dominated to non-dominated solutions gives a feel for how efficiently a method produces solutions that are worth investigating further (versus solutions that are dominated and whose discovery represents a waste of time).

N. Trial Protocol

The total amount of time allotted for each group was three hours. One hour was spent learning about the EFT model and the design tools while the remaining two hours were devoted to the design sessions. The first hour was basically the same for both groups in terms of procedure with the major differences emerging later during the sessions themselves. The trial introduction was presented with information about the trial purpose and a summary of the task. Background information was provided about the external fuel tank and the model to be used during the design session. A short description of the module and a simplified N^2 chart helped all participants know what information each module has as inputs and outputs and demonstrated the overall flow of information. Trial participants were also given a short introduction to the use of ICEMaker. The instruction focused on client usage and transferring data with the ICEMaker server since this is the primary skill required for the actual task.

Table 1. (left) Live Trial Schedule, (middle) control group composition, (right) test group composition

Live Trial Schedule	Control Group Composition	Test Group Composition
0:00–0:30 Trial introduction, purpose, and objectives 0:30–1:00 ICEMaker tutorial, EFT demo, trial goals and procedure 1:00–3:00 Design session and trade space exploration (additional design session optional) 3:00 on Post-trial debriefing and evaluation	Systems Engineer Structures Chair Aerodynamics Chair Cost Engineer	Systems Engineer Structures Chair Aerodynamics Chair Cost Engineer Optimization Chair

Participants were not given full information about the specifics of this research in order to preserve objectivity and reduce bias. For example, participants were aware of the existence of other groups but not of other methods. Further, no results from the design sessions were shared between the two groups until after the conclusion of both trials. It was necessary to explicitly emphasize to both groups that their design sessions should not be treated as a race or competition. The last part of the introduction was a trial run through the full procedure so that participants could gain a small amount of hands-on experience using the tools before beginning the design session. At the conclusion of the first hour, the team was given the green light to begin its independent evaluation of the EFT trade space and the clock was started.

Control Group Procedure

The control group requires four participants, one for each of the four EFT modules (structures, aerodynamics, cost, and systems). The procedure for the control group is somewhat simpler than for the optimization group. Without the complication of having to generate neural networks and run GAs, the control group design session is simply a standard ICEMaker session:

1. The structures chair modifies the input vector until he finds one that he or she believes is a good candidate. The chair then confirms that the selected vector meets all constraints on volume, stress, and vibration. If it passes, then the structures chair outputs his information to the ICEMaker server. If not, then the design vector must be tweaked until all constraints are met.

2. The aerodynamics and cost chairs request the latest information from the server and examine the effects the chosen vector has on their subsystems. In the absence of changes to the internal parameters of the model, these chairs' primary job is to make observations about the results of each change and to try and discern a pattern for what leads to a good tank design. This information should feed back to the structures chair after each iteration cycle. Once the two chairs have finished making their observations, they output their data to the ICEMaker server.

3. The systems chair requests the latest information from the server and adds the new design to the running table of discovered solutions. The visual depiction of the current design is automatically updated. The new point is also plotted on the performance-versus-cost chart and compared to previous solutions. From this information and the input of the other subsystem chairs, a new input vector can be devised and the cycle iterates until the Pareto front is well established or time expires.

Optimization Group Procedure

The optimization group requires five participants. With five participants, the setup is the same as for the control group with the extra participant placed in charge of the optimization module. With access to the optimization module, the optimization group follows a different procedure for its trial. It consists of a series of nested iterative loops:

1) At the beginning of the design session, the three main EFT modules (structures, aerodynamics, and cost) call up the optional optimization sheet within their ICEMaker client and initiate the neural network generation process. This takes approximately ten minutes. Once the process is complete, the NN data is saved in a common folder.

2) At this point, the design team breaks off into the foreground and background processes described in Section II.

a) The conventional design team (foreground) begins exploring the trade space as described for the control group.

b) Simultaneously, the optimization chair (background) collects the neural network data from the EFT model and uses the data to initiate a system-level optimization using a genetic algorithm.
3) Once the GA is finished and post-processing is completed, the optimization chair communicates the results to the rest of the team. Predicted Pareto-dominant points are tabulated and provided to the structures chair for evaluation using the full EFT model.
4) Steps 2 and 3 can be repeated. The foreground process investigates the new points discovered by the GA while the background process begins a new optimization run using different parameters (possible choices for GA parameter modification include population size, number of generations, cross-over probability, and mutation probability). Due to the stochastic nature of GAs, this parameter modification frequently results in the discovery of other Pareto-dominant solutions and allows the team to explore different parts of the trade space. It is important to note that step 1 never needs to be repeated unless some of the internal parameters of the EFT model are changed. The time spent generating the neural networks should be seen as a one-time investment that provides the design team with information about the trade space at the very beginning of the trial and continues to pay off periodically throughout the session.

O. Evaluation Metrics

The experimental framework described above as applied to evaluating different design methods is, to the author's knowledge, unique. Previous papers on new design methods certainly demonstrate their power and utility, but no direct comparisons using an experimental setup are used. It is usually left to the reader to attempt to quantify the differences between the methods, but in the absence of a controlled environment this data is difficult to obtain. This paper creates a first-cut attempt at developing such an environment. It is necessary to develop metrics by which the ISLOCE method can be compared to conventional design methods. Since no previous trials are available, a series of possible performance measures will be described below. Both the metric and the reasoning behind it will be given. It is hoped that future design methods will make use of them for their comparisons.

Independent metrics

These metrics can be used to quantify the stand-alone performance of a design method.
1. Maximum/minimum objective values – These values (located at the anchor points) come from the designs that have the 'best' possible value for a single objective, i.e. the global optima. In the EFT model, the key values are maximum available payload and minimum cost subject to meeting all constraints (Eq.14-16).
2. Raw number of point designs – This metric counts the total number of unique viable point designs developed by a method. While this gives no information about the quality of point designs (a random input vector generator would conceivably have a very high score), it provides a general feel for how long a method takes to create a viable solution. Combined with other metrics, this score can be used to determine average process loop time.
3. Raw number of "Pareto" optimal designs – This metric counts the number of unique point designs that are non-dominated when compared to all the solutions generated by a method. This metric also requires context to be interpreted correctly. It is easy to create a splash of points with a fraction of non-dominant ones among them if none of them are close to the true Pareto front. However, this metric serves as a measure of productivity as a method explores the trade space. A larger number of predicted non-dominated points means a greater number of points can be recommended for higher fidelity examination.
4. Ratio of dominated to non-dominated designs – This ratio provides a measure of method efficiency. A higher ratio implies less time is wasted discovering dominated solutions.
5. Normalized minimum Pareto front / utopia point distance – Given a spread of point designs, the trade space is normalized by placing the anchor points at opposite corners (1,0) and (0,1) with the nadir and utopia points defined as (0,0) and (1,1), respectively (for a two-objective maximization problem). This metric is defined as the shortest distance between a point design on the Pareto front and the utopia point. The point chosen on the Pareto front must be an actual design that satisfies all constraints.

Comparative metrics

These metrics are best used to compare the relative performance of two different design methods.
1. Anchor point spread – This metric is given as the range of objective values defined by the anchor points. It is a measure of how completely a method explores a trade space within specified side constraints on the design vector.
2. Ratio of cross-dominated solutions – This metric takes the Pareto front of one method and counts the number of dominated Pareto front designs generated by the other method. The ratio of these two counts provides a measure of how much more effective one method was over the other in approaching the true Pareto front. A ratio close to

one implies both methods are relatively equal in performance (or explored disparate areas of the trade space). A ratio far from unity implies one method was significantly better at getting close to the true Pareto front.

P. Trial Results

The live test with the setup and procedures listed above was conducted in May of 2004 in the MIT Department of Aeronautics and Astronautics design studio (33-218). Eight MIT Aero/Astro graduate students were recruited to participate in the trial. The students were selected based on availability, familiarity with the EFT model, past experience with concurrent engineering, knowledge of optimization basics, and personal interest in the project. Team assignments were made according to Table 1 (middle, right); individual participants are acknowledged below. The control group trial was conducted first, then the optimization group trial a week later. No information about the results of the trials was shared between groups until after the conclusion of the second test. The results of the live tests will be presented one group at a time and then combined in an additional section for comparative analysis.

Control Group Performance

The control group's performance set the baseline for evaluation of the ISLOCE method. With no access to optimization, the only trade space knowledge the group started with was a single data point: the nominal values of a standard tank. The control group's initial approach was to make small perturbations to the original design vector and examine the effects on the tank's performance. During this stage, most changes were made to the tank's geometric dimensions with only secondary consideration paid to the component thicknesses. The result was a series of designs that became progressively cheaper, but could not carry a significant amount of payload due to the added weight from excessive material thickness. Later, as the team gained more knowledge of the interaction of various parameters, they became more adept at modifying multiple parameters at a time. They learned how to tune the component thicknesses to the minimum values allowed by the constraints in order to achieve the lightest design possible for a given set of geometric dimensions. This knowledge led them to the high payload/ high cost region of the trade space. Towards the end of the design session, the control group progressed back to the nominal design regime and completed their exploration of the trade space in the low payload / low cost region. The detailed results of the control group's design trajectory are contained in Appendix A. Note that point 9 is significantly worse than all other designs and is not shown on the plot (Fig.14) for scaling reasons. The majority of solutions found by the control group are arranged in a fairly linear pattern between the two anchor points. With the exception of two outliers, there is very little scattering of points away from the predicted Pareto front. No point was found which dominated the nominal (starting) solution. The performance metrics for the control group are summarized in Table 2.

Table 2 – Control group performance summary

Min/Max objective values	max payload = 35,948 kg min cost = $449,640
Number of point designs	26 viable designs, or roughly 13 per hour
Number of "optimal" designs	10 non-dominated designs (including the nominal point)
Ratio of dominated to non-dominated solutions	10/26 or ~ 38%
Normalized minimum utopia point distance	closest Pareto point to utopia: design 19 (0.538, 0.591) => 0.617 from the point (1,0)
Anchor point spread	payload: {19221,35948} => 16727 cost: {449640, 567545} => 117905

Optimization Group Performance

The optimization group's performance benefited significantly from access to optimization through the ISLOCE method. Although the team's progress was somewhat haphazard at times, the overall results were an improvement over the baseline established by the control group. The group had no problems with generating the neural networks for the optimization chair (Section I) and running genetic algorithms (Section J) during the design session. This represented an initial investment and meant that the progress of this group was initially delayed relative to the control group.

The exploration in parallel by the optimizer and the human team worked as predicted. The optimization chair and the rest of the team complimented each other's work by supplying each other with data on the solutions their respective methods developed. Progress for the optimization "group" came in waves as the optimizer provided new sets of points for exploration by the rest of the team. Due to the stochastic nature of genetic algorithms, some predicted Pareto fronts actually resulted in dominated designs and thus wasted time. *However, the GA was also able to point the optimization group towards regions of the trade space that the control group did not find.* The numerical results of the optimization group trial are listed in Appendix B (Fig. 15), in the order in which they were discovered. Note that design 28 is much worse than the other designs and is not shown on the plot for scaling reasons.

The results of the optimization group *display a very different pattern* from those generated by the control group. Whereas the control group's data showed a linear distribution, the optimization group's solutions are far more scattered across the trade space. This effect is due mainly to the team's investigation of GA-supplied points that turned out to be dominated when evaluated using the full EFT model. In terms of performance, the optimization group was able to find points that dominated the nominal point provided at the start of the trial, as well as most of the points found by the control group. The performance metrics for the optimization group are summarized in Table 3.

Table 3 – Test group performance summary

Min/Max objective values	max payload = 37,181 kg min cost = $471,825
Number of point designs	33 viable designs, or roughly 17 per hour
Number of optimal designs	7 Pareto designs
Ratio of dominated to non-dominated solutions	7/33 or ~ 21%
Normalized minimum utopia point distance	closest Pareto point to utopia: design 25 (0.797, 0.595) => 0.453 from the point (1,0),
Anchor point spread	payload: {20548,37181} => 16633 cost: {471825, 554732} => 82907

Q. Combined Results and Interpretation

The combined results of both trials provide a great deal of information, not only on the effectiveness of the two methods but also on the benefits and issues associated with using optimization in a concurrent engineering environment. The two primary sources for this information are the combined results of the tradespace exploration and the comparison of metrics established in the two trials. While most of the data gleaned from the trials is quantitative, it is equally important to investigate the qualitative data produced by the trials, specifically from comments made by participants during the trial. As will be shown, these less-tangible features of the two approaches can significantly contribute to a method's performance.

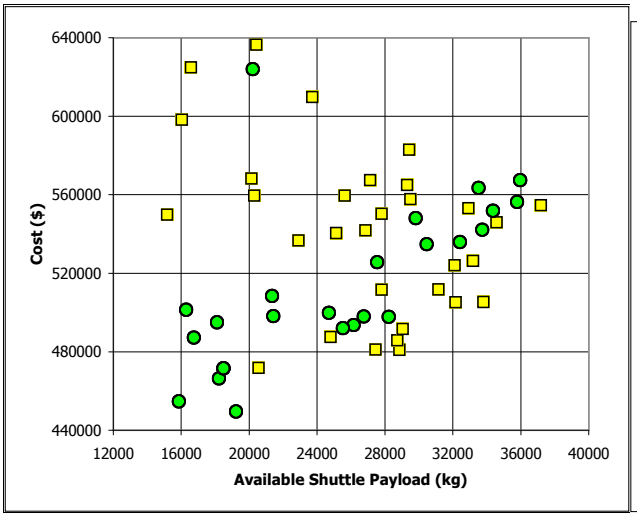


Figure 12. Combined Results: ○=ctrl group, □= test group

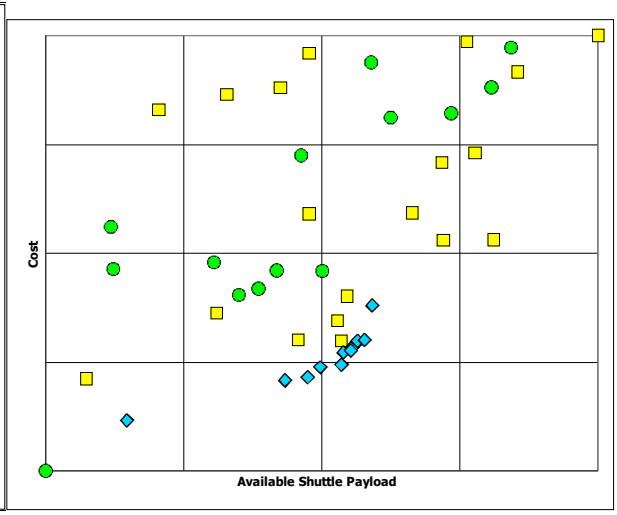


Fig.13. Normalized Results: ○=ctrl , □= test grp, ◊ = GA

The combined results of the two trials are shown in Figures 12 and 13, respectively. Figure 12 plots the data from Figures 14 and 15 on the same axes for the purpose of direct visual comparison of the regions explored by both methods. In terms of finding near-Pareto optimal designs, the optimization group's results (□) can clearly be seen to dominate those of the control group (○) over most regions of the tradespace. The control group does a slightly better job in the low payload / low cost region of the tradespace, but it still has no points that dominate any of the points of the optimization group "Pareto" front. The price the optimization group pays for this increase in performance is also visible in the chart. The optimization group's points are scattered throughout the tradespace, with the vast majority of points dominated by other designs. This scattering represents 'wasted' time and effort by the design team, although this time can also be seen as an investment or cost for obtaining higher performing solutions.

Figure 13 shows the same data as the previous figure, only normalized against the two best anchor points found during the trials. Also shown are points (◊) obtained by running a GA autonomously on the model, offline after the human trials. The offline GA found points that dominated those of both groups, but not over the entire trade space. The suspected problem is a phenomenon called "niching" where points tend to cluster during multiobjective optimization with genetic algorithms. No anchor points could be found that were superior to the ones found by the design teams. The important thing to notice is that the 'true' Pareto front is relatively near the one predicted by the control group and very near the front predicted by the optimization group.

The performance of the two groups can be examined further by comparing the methods based on the metrics developed earlier. A summary of this comparison is detailed in Table 4.

Table 4 – Comparison of group design performance

	Control Group	Test Group	% Improvement
Min/Max objective values			
Maximum payload	35,948 kg	37,181 kg	3.4
Minimum cost	$449,640	$471,825	(- 5.0)
# of point designs	26	33	26.9
# of non-dominated designs	10	7	(- 30.0)
Ratio of dominated to non-dominated designs	38%	21%	(- 44.7)
Normalized minimum utopia point distance			
intra-method	0.617	0.453	26.6
overall	0.678	0.563	17.0
Anchor point spread			
payload	16727	16633	(- 0.6)
cost	117905	82907	(- 29.7)

These results help illustrate in more detail the visual results from Figures 12-15. The optimization group was able to locate the highest payload solution while the control group found the point design with the lowest cost. The scale of these differences is relatively small, but in the aerospace industry, a 3.4% boost in payload or a 5% reduction in cost can have a major impact on the viability of a program. The optimization (=test) group was able to develop about 25% more viable point designs than the control group. The optimization team was able to use many points directly from the optimization run while the control group was forced to develop all of their designs independently. However, the majority of these points were poor overall designs. The control group had nearly double the ratio of dominated to non-dominated solutions compared the optimization group. Much of the optimization group's time was spent evaluating supposedly good points recommended by the GA, only to find that most of them were dominated solutions. The payoff from this 'wasted' time however is seen in the next metric. The optimization group did a much better job at pushing their solutions closer to the utopia point. The best 'true' minimum distance found during trial post-processing was 0.520, 7% better than the best value found by the optimization team. It should be noted that this best value was the result of a genetic algorithm (◊) with many times more individuals and generations than the one run by the optimization team, and consequently ran for hours rather than minutes. Even more impressive is the fact that the optimization team was not even using the full EFT model but instead used a series of approximations. While these results are only from a single test, if the ISLOCE method is capable of consistently matching within 10% the performance of a full GA in a fraction of the time, its application in industry could lead to a significant increase in the productivity of conceptual design teams.

V. Conclusions

R. Benefits and Impact

This paper introduces a new method that attempts to unify multidisciplinary optimization with problem decomposition in an integrated concurrent engineering environment. Known as ISLOCE (for 'Integrated System-Level Optimization for Concurrent Engineering'), this method has the potential to put the power of modern system-level optimization techniques in the hands of engineers working on distributed problems while retaining the speed and efficiency of concurrent engineering practices.

A parallel optimization approach to concurrent engineering could offer great benefits, any large scale project that makes use of conceptual design. Traditionally, optimization has been conducted by a small group of people separated from the rest of the design team. Their models are extremely complex and may take days to run. The results from such a team are useful but are not flexible enough to handle the rapid model changes that often occur during concurrent engineering. This might be one of the reasons why full-scale MDO techniques have had difficulty being infused into the mainstream design processes of major organizations. The parallel approach presented here brings the optimization team right into the design studio, allowing them to directly impact a design in real time, while interacting with non-optimization disciplinary specialists. The ability to quickly identify a set of interesting candidate solutions and guide an engineering team towards them will have a significant impact on the efficiency of design sessions.

S. Limitations

It is important to note that ICEMaker is not an all-in-one automated spacecraft or aircraft generator, nor is it a high-end symbolic calculation tool. It simply serves as a compliment to the ICE method by enabling multidisciplinary trades through parameter sharing. The end designs developed using ICEMaker are only as accurate as the models they are based on. With this in mind, there are many problems that are unsuitable for ICEMaker usage in the context of the framework presented here. Typically, models for ICEMaker clients are developed with Excel or with software that is easily linked to Excel such as Matlab. CAD or finite-element programs are more difficult to interface. Furthermore, the data that can be transferred through ICEMaker is limited to those formats capable of being expressed in an Excel cell, typically real numbers or text strings. Approximate geometry, timelines, and other qualitative information are very difficult to express in this way. ICEMaker is most powerful for tackling highly quantitative problems with well-defined interfaces between subsystems. Recognizing both the potential and the limitations of ICEMaker is essential for proper usage.

T. Future Work

The work presented here is of a preliminary nature, both in terms of formulation and implementation. Ideally, the optimization sheet would be general enough to be included as part of the basic initialization of every ICEMaker client from now on. Design teams could use it as needed, but its inclusion in the client would have no effect if optimization were not required. We will continue research in parallelism between system optimization and ICE in the future. These activities will focus on the following areas:

1. Refinement of optimization client and optimization chair implementations
2. Comparison of CO, BLISS and ISLOCE for a set of benchmark problems
3. Application of ISLOCE to an industrial strength problem in a professional organization (e.g. JPL, General Motors …) to obtain feedback from professional engineers
4. Refinement and test of other background optimizers such as Simulated Annealing or a self-tuning GA
5. Better computation of GA-generated Pareto Front for EFT case, e.g. via mating restrictions
6. Repetition of the live trials over a larger set of groups to ascertain the statistical validity of the results
7. Allow matrix decomposition of systems according to both disciplinary and subsystem dimensions

Appendix A – Results obtained by Control Group (ICE without System-Level Optimization)

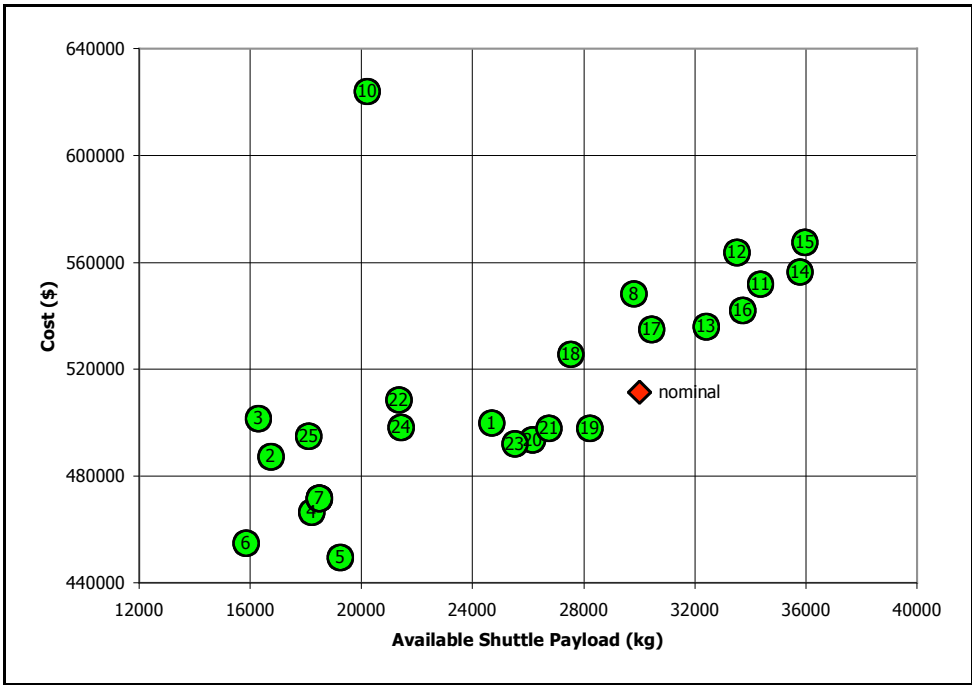


Figure 14. Objective Space Results generated by the control group

Table 5 – Design trajectory for control group

Design #	Payload	Cost		Length	Radius	t_cy	t_s	t_co	h/R
nominal	30000	$511,424		41.5	4.50	0.0070	0.0080	0.0075	1.0
1	24688	$500,018		40.0	4.60	0.0070	0.0090	0.0090	0.5
2	16727	$487,343		30.0	5.20	0.0090	0.0100	0.0100	0.5
3	16285	$501,558		26.0	5.50	0.0100	0.0100	0.0120	1.0
4	18205	$466,533		22.0	5.80	0.0090	0.0110	0.0100	1.0
5	19221	$449,640		20.0	5.90	0.0090	0.0105	0.0095	1.0
6	15844	$454,868		18.0	6.15	0.0095	0.0110	0.0100	1.0
7	18475	$471,703		16.0	6.15	0.0095	0.0110	0.0095	2.0
8	29800	$548,232		45.0	4.25	0.0080	0.0080	0.0070	1.5
9	2219	$716,579		40.0	4.50	0.0150	0.0150	0.0150	1.5
10	20204	$624,032		50.0	4.10	0.0100	0.0100	0.0100	1.5
11	34351	$551,885		50.0	4.10	0.0065	0.0080	0.0065	1.5
12	33509	$563,704		51.5	4.10	0.0065	0.0080	0.0065	1.5
13	32409	$535,983		45.0	4.30	0.0070	0.0080	0.0070	1.5
14	35773	$556,479		45.0	4.20	0.0065	0.0075	0.0065	3.0
15	35948	$567,545		45.0	4.15	0.0065	0.0075	0.0065	3.6
16	33712	$542,187		40.0	4.40	0.0070	0.0080	0.0070	3.0
17	30437	$534,968		35.0	4.70	0.0075	0.0085	0.0075	3.0
18	27524	$525,780		30.0	5.00	0.0080	0.0090	0.0080	3.0
19	28216	$497,903		25.0	5.20	0.0080	0.0095	0.0080	3.0
20	26148	$493,678		22.0	5.40	0.0085	0.0095	0.0085	3.0
21	26736	$498,049		23.5	5.28	0.0085	0.0095	0.0085	3.0
22	21344	$508,555		20.0	5.70	0.0090	0.0100	0.0090	3.0
23	25507	$492,148		21.0	5.48	0.0085	0.0100	0.0085	3.0
24	21416	$498,273		18.0	5.80	0.0090	0.0105	0.0090	3.0
25	18093	$495,035		14.0	6.15	0.0095	0.0110	0.0095	3.0

Appendix B – Results obtained by Test Group (using ISLOCE method)

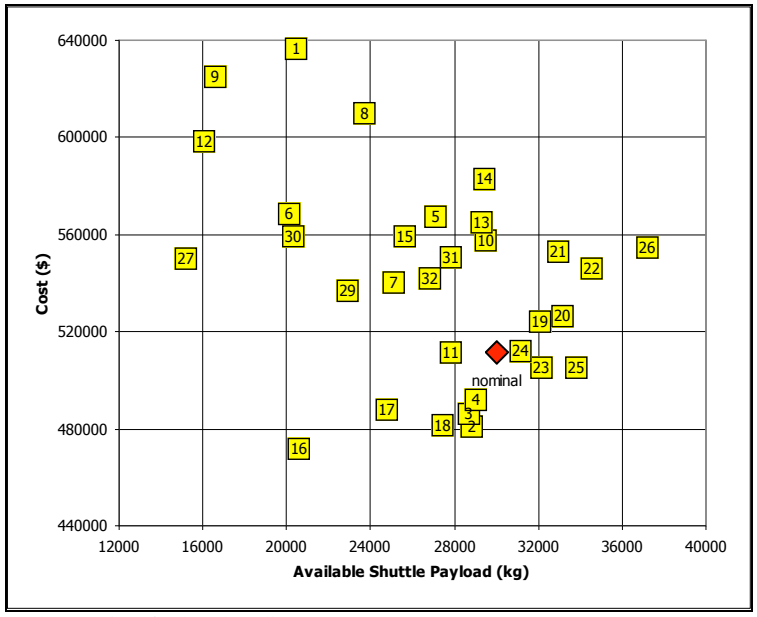


Figure 15. Objective Space Results generated by the test group

Table 6 – Design trajectory for test group

Design #	Payload	Cost		Length	Radius	t_cy	t_s	t_co	h/R
nominal	30000	$511,424		41.5	4.50	0.00700	0.00800	0.00750	1.0
1	20414	$636,561		50.00	4.10	0.01000	0.01000	0.01000	2.00
2	28836	$481,042		30.43	5.00	0.00790	0.00890	0.00790	1.54
3	28709	$485,913		29.70	5.05	0.00790	0.00890	0.00790	1.79
4	29036	$491,730		30.40	4.99	0.00790	0.00890	0.00790	1.92
5	27113	$567,546		50.00	4.10	0.00790	0.00890	0.00790	1.00
6	20132	$568,309		20.00	5.30	0.01000	0.01000	0.01000	5.00
7	25116	$540,505		20.00	5.30	0.00850	0.00940	0.00850	5.00
8	23708	$609,920		32.00	4.68	0.00720	0.01000	0.01400	4.97
9	16566	$624,993		27.50	4.90	0.01000	0.01000	0.01400	4.78
10	29484	$557,806		32.70	4.70	0.00720	0.00870	0.00910	4.29
11	27802	$511,695		32.70	4.90	0.00800	0.00870	0.00910	2.00
12	16026	$598,294		55.00	4.00	0.00800	0.00870	0.01400	0.25
13	29298	$565,132		30.75	4.67	0.00710	0.00890	0.01000	4.99
14	29416	$583,020		35.00	4.50	0.00710	0.00890	0.01000	4.99
15	25626	$559,545		25.00	5.00	0.00800	0.00890	0.01000	4.99
16	20548	$471,825		25.00	5.50	0.00900	0.01000	0.01000	1.00
17	24789	$487,696		25.00	5.40	0.00850	0.00950	0.00900	2.00
18	27432	$481,221		30.58	5.00	0.00780	0.00990	0.01000	1.32
19	32105	$524,101		38.52	4.60	0.00700	0.00860	0.00700	2.16
20	33180	$526,384		39.73	4.50	0.00690	0.00860	0.00685	2.23
21	32918	$553,254		45.00	4.30	0.00690	0.00860	0.00685	2.23
22	34570	$545,937		45.00	4.30	0.00655	0.00755	0.00655	2.23
23	32153	$505,230		40.00	4.50	0.00690	0.00800	0.00830	1.30
24	31151	$511,836		42.00	4.48	0.00680	0.00785	0.00695	1.07
25	33800	$505,394		40.00	4.47	0.00680	0.00785	0.00695	1.53
26	37181	$554,732		50.00	4.05	0.00615	0.00710	0.00615	2.10
27	15173	$549,933		10.00	6.20	0.00940	0.01090	0.00950	5.00
28	1181	$539,942		10.00	7.00	0.01200	0.01400	0.01400	2.00
29	22908	$536,792		16.55	5.60	0.00850	0.00980	0.00850	4.97
30	20304	$559,528		18.43	5.41	0.00820	0.01100	0.01200	4.83
31	27797	$550,454		44.70	4.30	0.00810	0.00870	0.00880	1.32
32	26854	$542,026		50.00	4.21	0.00640	0.00820	0.00880	0.51

Acknowledgments

The authors thank the following graduate students from MIT for their participation in the live trials: Gergana Bounova, Babak Cohanim, Masha Ishutkina, Xiang Li, William Nadir, Simon Nolet, Ryan Peoples, and Theresa Robinson. This research was supported by the MIT Department of Aeronautics and Astronautics through a CDIO Teaching Assistantship. Feedback was provided by Prof. David Miller, Prof. Jeffrey A. Hoffman, Col. (ret) John Keesee, and Col. (ret) Peter W. Young. The first author also acknowledges support by his former advisor at Caltech, Dr. Joel Sercel, who is also credited as the inventor of the ICEmaker software and method.

References

[1]Sobieszczanski-Sobieski, J.; Barthelemy, J.-F. M.; and Giles, G. L.: "Aerospace Engineering Design by Systematic Decomposition and Multilevel Optimization," 14-th Congress of the International Council of the Aeronautical Sciences (ICAS), Proceedings of; Toulouse, France, 1984

[2]AIAA Technical Committee on Multidisciplinary Design Optimization, White Paper on Current State of the Art, Jan. 1991.

[3]Caltech Laboratory for Spacecraft and Mission Design homepage: http://www.lsmd.caltech.edu

[4]Sercel, J., "ICE Heats Up Design", *Aerospace America*, July 1998

[5]Parkin, K., Sercel, J., Liu, M., and Thunnissen, D., "ICEMaker: An Excel-Based Environment for Collaborative Design," 2003 IEEE Aerospace Conference Proceedings, Big Sky, Montana, March 2003.

[6]Braun, R.D., "Collaborative Optimization: An Architecture for Large-Scale Distributed Design", Ph.D. Dissertation, Stanford University, May 1996.

[7]Braun, R.D., Gage, P.J., Kroo, I.M., Sobieski, I.P., "Implementation and Performance Issues in Collaborative Optimization", Sixth AIAA/USAF MDO Symposium, Bellevue, WA, AIAA-94-4325, Sept. 1996.

[8]Kroo, I.M., Sobieski, I.P., "Collaborative Optimization Using Response Surface Estimation", AIAA #98-0915, 1993.

[9] Sobieszczanski-Sobieski, J., Agte, J.S., Sandusky, R.R., "Bilevel Integrated System Synthesis (BLISS)", NASA/TM-1998-208715, August 1998b.

[10] Sobieszczanski-Sobieski, J., Emiley, M.S., Agte, J., Sandusky, R., Jr. "Advancement of Bi-level Integrated System Synthesis (BLISS)", AIAA 2000-0421, AIAA 38 th Aerospace Sciences Meeting, Reno, Jan. 2000.

[11]Sobieszczanski-Sobieski, J., Altus, T., Phillips, M., Sandusky, R., "Bi-Level Integrated System Synthesis (BLISS) for Concurrent and Distributed Processing", 9 th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA., AIAA 2002-5409, Sept. 2002.

[12]Jackson, D. The Theory of Approximation. New York: American Mathematical Society, p. 76, 1930.

[13]Demuth, Howard and Beale, Mark, Neural Network Toolbox for Use with MATLAB. The MathWorks, Inc., 1998.

[14] Goldberg, David E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, 1989.

[15]Kodiyalam, S., Sobieszczanski-Sobieski, J., "Bi-Level Integrated Systems Synthesis with Response Surfaces", 40 th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, St. Louis, MO., Apr. 1999. AIAA 99-1306-wip

[16] Sobieszczanski-Sobieski, J. "Different Objectives Result in Different Designs". AIAA MDO Short Course, Atlanta, GA, Sept. 2002.