SHIP DESIGN USING HEURISTIC OPTIMIZATION METHODS

Robert Wolf
John Dickmann
Ryan Boas
Massachusetts Institute of Technology

ABSTRACT

Preliminary ship design is currently more art than science, heavily dependent on highly experienced naval architects. The use of genetic algorithms is proposed as a method for improving ship design through more effective exploration of the design space. Three main points are advanced in this paper. First, genetic algorithms (GA) are a highly effective tool for the exploration of large-scale, nonlinear design spaces and, when combined with gradient based search techniques, may provide a more computationally efficient means of identifying near optimal designs. Second, GA methods may provide a high utility tool that can enhance the ship design process. Third, the current design choice method of weighted objective measures of effectiveness can mask potentially useful areas of the design space.

INTRODUCTION

Ship design is a complex endeavor requiring the successful coordination of many different disciplines, both technical and non-technical. Preliminary design is the least defined stage of the ship design process and seeks to define the basic payloads and ship size characteristics. It begins with highly experienced decision makers and end-users who attempt to articulate their desires and the tradeoffs they are willing to allow. This process is more art than science as decision makers must select design goals without a complete understanding of the effect of those goals on the final design. Using these goals, experienced naval architects work through design trades in an informal manner, relying primarily on their experience to guide major design parameters. With several designs created in this manner, a selection is made by comparing estimated costs versus designed performance. Once preliminary design has been accomplished, the more formalized process of iterative design can begin.

This paper builds on previous work[1] to further extend the use of formalized optimization methods in preliminary ship design. The goal of this effort was to examine methods for exploring a design space currently explored by experienced designers using heuristics—tacit knowledge gained through a process of trial and error, often over the course of years. Selection of designs and design goals by experts is highly dependent on specific individuals involved in the process, more specifically, their personal and professional experiences. Tools that help make the outcome of the preliminary design process more consistent would be invaluable. These tools are meant to augment, rather than replace the skills of the highly experienced decision maker.

An existing simulation code, the MIT Math Model, was selected as the modeling core. This model is primarily empirical, developed to perform preliminary mono-hull warship design. Both gradient search and genetic algorithms were wrapped around the simulation code core in order to explore the highly nonlinear trade-space associated with ship design.

This paper provides a brief discussion of the MIT Math Model simulation code, followed by an initial trade-space exploration. Next, gradient search methods are used to explore the trade-space, followed by genetic algorithms. The paper closes with a conclusions and future work section that highlights the opportunities for further investigation into formalized methods of preliminary ship design.

MODEL IMPLEMENTATION

The MIT Math Model (MMM) is a primarily empirical model, originally developed in 1975 by Graham and Hamly. It is based on data obtained from frigates and destroyers built since the end of World War II and is applicable to the design of 2500 to 8500 ton mono-hull surface combatants using standard Navy design practices. The model's primary purpose was, and still

[†] This paper is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

is, to serve as an educational tool for naval architects that is “a balance between simplicity, sophistication / completeness, generality, and compatibility with computers.” The MMM has gone through numerous revisions since its original inception with the most recent being a conversion to MATLAB by Wolf, Riyadi and Sudoff[1]. The MATLAB version of the MMM serves as the core model upon which this optimization work is built.

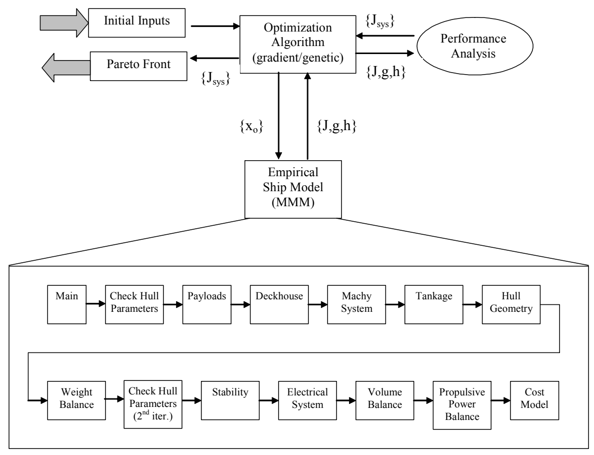


Figure 1: Block diagram

As shown in Figure 1, the optimizer is linked to the empirical ship model (MIT Math Model). In the case of the GA implementation, it is also linked with a performance analysis block that determines the fitness of each design. The various routines used in the MIT Math Model are described below.

- **Main-**Calls each of the subroutines in order
- **Payloads-**Summarizes the weights, areas, vertical centers of gravity (VCG), and electrical power requirements for a specified payload choice
- **Deckhouse-**Determines the minimum deckhouse size.
- **MachySystem-**Determines the machinery box size and the required inlet and exhaust area
- **Tankage-**Determines the required tank (e.g. fuel, potable water, sewage)
- **HullGeometry-**Determines the required hull area
- **WeightBalance-**Determines total and individual group weights
- **CheckHullParameters-**Determines the hull depth at various stations and ship draft
- **Stability-**Calculates the ship VCG, metacenter, center of buoyancy, and determines stability coefficient
- **ElectricalSystem-**Determines the electrical load
- **VolumeBalance-**Determines the ship volume and deck area available
- **PropulsivePowerBalance-**Determines the maximum speed and endurance range
- **CostModel-**Determines the cost from empirical formulas based on weight

Equation (1) describes the optimization formulation mathematically. The constraints ensure the variables are within acceptable bounds, the ship has sufficient power and size to meet requirements, the ship is stable at sea and that it does not go too far outside the bounds of the empirical model. The key variables used within the ship model along with their upper and lower bounds are included in Tables 1 and 2.

$$\max \mathbf{J(x)} = [\text{-Cost, Speed, Range, Combat Capability}]^T$$

$$\begin{aligned}\text{s.t. } & \mathbf{x_{lb}} \le \mathbf{x} \le \mathbf{x_{ub}} \\ & \text{Electrical Power Error} \ge 0 \\ & \text{Area Error} \ge 0 \\ & \text{Volume Error} \ge 0 \\ & .09 \le \text{Stability Measure} \le .122 \\ & 6.5 \le \text{Length-to-Beam Ratio} \le 9.5 \\ & 45 \le \text{Displacement-to-Length Ratio} \le 65 \\ & 2.8 \le \text{Beam-to-Draft Ratio} \le 3.7 \qquad (1)\end{aligned}$$

As with any model, especially of a complex system, there is a trade between fidelity and execution time. Examination of a large number of solutions is highly desirable, especially in the case of a design space that contains discrete variables. The MIT Math Model is an extremely efficient model, allowing very high numbers of “design experiments” to be executed.

Table 1. Continuous Input Variables of the MIT Math Model

Continuous Variables	**Min Value**	**Max Value**
Length (ft)	250	600
Beam (ft)	10	75
Cp (Prismatic Coefficient)	0.54	0.64
Cx (Max Section Coefficient)	0.7	0.85
Hull deck height (ft)	8	12
Deckhouse deck height (ft)	8	12
Fraction of min station 0 depth (ft)	1	1.5
Fraction of min station 10 depth (ft)	1	1.5
Fraction of min station 20 depth (ft)	1	1.5
Bilge height (ft)	1	10
KG margin	0.5	1
Volume factor	3.5	5.2
Fuel (lton)	10	600

Table 2. Discrete Input Variables of the MIT Math Model

Discrete Variables	# Options
# of hull decks	1-4
# of deckhouse decks	1-4
Engine Choice	7
Engine-Prop Configuration	4
Generator Choice	7
# of generators	2-4
Payload Choice	48
Deckhouse Material (A1,Steel)	2
Collective Protection System	T/F

MODEL VALIDATION

To validate this model, an actual US frigate design, the FFG-7, was used. Data was obtained for the ship from ASSET, a standard US Navy warship design tool, including ship hull dimensions, generator type and number, as well as other ship characteristics. Since the MIT model generator choices were all more than 50% larger than the one used on the FFG-7, the parameters for the FFG-7 were entered as a new generator choice. The payload of the MIT model was not identical to the FFG-7, but major components such as a helicopter and large weapons were included in the validation model. These inputs, shown in Table 3, were entered into the MATLAB model; comparisons of results are shown in Table 4. The largest error, range, is due to the specific fuel consumption calculation which assumes no dependence on output power thus overestimating range which is calculated at non-optimal specific fuel consumption. The remaining errors can also be explained due to model simplifications and are within accepted tolerances for this stage of design.

Table 3. Model Validation Inputs for FFG-7

Inputs (Design Vector)	Value
Length (ft)	408
Beam (ft)	44.8
Prismatic coefficient	.6
Maximum section coefficient	.75
Number of hull decks	2
Number of deckhouse decks	3
Average hull deck height (ft)	9.77
Average deckhouse deck ht. (ft)	8.58
Bilge height (ft)	9.5
Engine choice	6
Engine/propeller combination	2
Generator choice	(Data entered)
Number of generators	4
Fuel Weight (lton)	550
Deckhouse mat'l (Al=1, steel=2)	2
KG Margin	1
Collective Protection System	None
Volume factor	3.5

Table 4. Model Validation Comparisons to FFG-7

Comparison Outputs	Model	FFG-7 Frigate	% Variation
Draft (ft)	15.91	15.69	1.40%
Station 0 Depth (ft)	42.28	39.54	6.93%
Station 10 Depth (ft)	29.04	30	-3.20%
Station 20 Depth (ft)	30.96	30.86	0.32%
Full Load Weight (lton)	3738	3935	-5.01%
Installed Power (hp)	50004	43000	16.29%
Ave kW Load (kW)	1244.9	1372.08	-9.27%
Ave kW Margin Load (kW)	2749.6	3382.95	-18.72%
kW Error	2.15%	?? (>0)	
Area Error	-0.29%	?? (>0)	
Volume Error	-.0.26%	?? (>0)	
Stability Measure (GM/B)	.0955	.086	11.05%
Cost ($M)	348.5		
Max Sustained Speed (kts)	31.5161	29+	~8.68%
Range (nm)	5628.2	~4200	~34.00%

INITIAL DESIGN SPACE EXPLORATION

Combinatorial and Design of Experiments (DOE) methods were utilized in order to better understand the ship design landscape. As stated in the previous section, the efficiency of the empirical ship model enables large-scale exploration of the trade space. (In compiled form, the model is able to check approximately 100 designs per second.) First, the MMM was run through a thorough combinatorial exploration of over 6.9 million design options, computing 53,897 feasible designs, a ratio of about 1:130. The full run was completed on a personal computer in approximately 20 hours.

Given the high number of infeasible designs and the clustering observed in the feasible designs, it is apparent that there are "design feasibility islands" surrounded by large areas of infeasible ship designs. (Refer to Figure 2.) As will become clear later in the report, these islands have significant impact on effectiveness of heuristic and formal mathematical optimization techniques. It should also be noted that within the feasibility islands, the design landscape is discontinuous due to the many discrete variables such as engine type, engine number and combat payload.

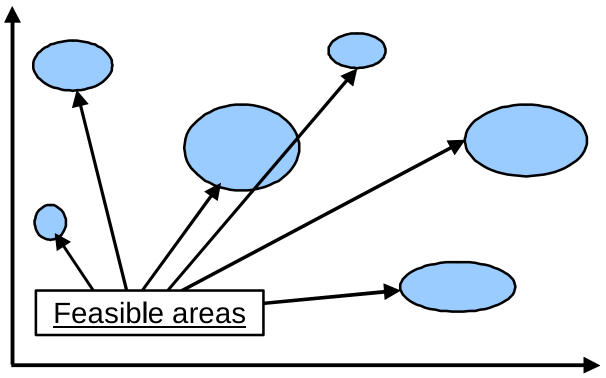


Figure 2: Visualization of the ship design space.

Table 5: Free design variables with upper and lower bounds.

Design Variable	Min value	Max value
Length	250	600
Beam[‡]	(38.5)	(63.2)
Prismatic coefficient	0.54	0.64
Maximum section	0.75	0.85
Fuel Weight	100	600

Table 6. Fixed Design Variables

Variable	Fixed Value
Number of hull decks	2
Number of deckhouse decks	3
Average hull deck height	9.77
Average deckhouse deck height	8.58
Fraction of min depth at Sta. 0	1
Fraction of min depth at Sta. 10	1
Fraction of min depth at Sta. 20	1
Bilge height	9.5
Engine choice	6
Engine/propeller combination	2
Generator choice	3
Number of generators	4
Payload choice	48
Deckhouse mat'l (Al = 1, steel = 2)	2
KG Margin	1
Collective Protection System	0
Volume factor	3.5

[‡] The values for Beam were determined using a length-to-beam ratio bounded by the constraint values.

GRADIENT SEARCH

As previously mentioned, gradient-based methods are useful for analyzing the continuous regions of a design space. Given this, the problem had to be significantly limited to a region of the design space with continuous variables only. Table 5 shows the continuous variable used and their bounds. Table 6 shows the input variables held fixed and their assigned values.

Sequential Quadratic Programming was used to test the effectiveness of gradient-based methods in exploring the restricted ship design space. The optimization was implemented using the MATLAB 'fmincon' function. The FFG-7 design was chosen as the initial design vector. Detailed analysis of the Hessian matrix indicated that scaling was an issue; thus all variables were scaled to O(0) using an interface between the optimization code and the simulation code. Post-scaling, the gradient optimization was very well behaved and was insensitive to the initial design vector, generating similar results regardless of starting point. The only problem with this method is in its inability to handle the model's discrete variables. The gradient-based optimum is shown in Table 7

Table 7: Gradient-based optimum

		Design Variables:					
	Time	L	L/B	Cp	Cx	Fuel	Spd
Gradient Optimization	2s	436	9.5	0.54	0.75	481	32.86

GENETIC ALGORITHMS

As previously noted, the ship design space is highly nonlinear. There are many discrete variables, such as engine type (power), propeller number and type and payload (determines combat capability). Gradient methods can only handle these types of variables by running separate optimizations for each discrete variable combination or by replacing discrete variables with continuous approximations and then rounding to a discrete variable after optimization. Performing separate optimizations for each set of discrete variables becomes extremely inefficient for large numbers of variables and rounding is limited by lack of a continuous gradient.

Heuristic methods are able to handle both discrete and continuous variables, making them well suited to large, multidisciplinary design problems. Genetic algorithms were selected over other heuristic methods because of their ability to handle discrete variables easily and because they can be used to develop a population of solutions useful in multi-objective optimization.

The Genetic Algorithm Optimization Toolbox (GAOT) was used as the starting point for the heuristic portion of the ship design space exploration. Default settings utilized initially included floating point genes with a normalized geometric selection[§].

Mutation was performed on a specified number of randomly selected individuals using four different operators – boundary (one gene mutated to a boundary value), uniform (one gene uniformly mutated between the boundaries), non-uniform (one gene modified by a Gaussian distribution from is original point)[**], and multi non-uniform (similar to non-uniform, but affecting all genes). The mutation rates presented in this paper are defined as the number of mutations per generation.

Crossover operations utilized three operators – arithmetic (a random interpolation between the parents), heuristic (an extrapolation of the vector formed by the two parents in the direction of the better parent), and simple (genes are exchanged between parents). All crossover rates are defined as the number of crossovers, regardless of total population.

The default mutation and crossover rates are included in Table 8 and Table 9.

Table 8: Default mutation rates

Boundary	Uniform	Non-Uniform	Muti-Non-Uniform
4	4	4	6

Table 9: Default crossover rates

Arithmetic	Heuristic	Simple
2	2	2

A flowchart for a basic genetic algorithm is shown Figure 3.

Figure 3: Flowchart of basic genetic algorithm
(Taken from GOSET 1.03 Manual)

Initial GA Results

As an initial check on GA performance, a population size of 80 was run for 100 generations, using the default values (including the design space limitations) listed in the previous section. The results represent the best design solution based on the remaining population at the end of the GA run and are almost identical to the gradient-based solution.

A side-by-side comparison of the genetic algorithm and gradient based solutions is provided in Table 10.

Table 10: Genetic Algorithm Performance Compared with Gradient Search (SQP)

		Design Variables:					
	Time	L	L/B	Cp	Cx	Fuel	Spd
Gradient Optimization	2s	436	9.5	0.54	0.75	481	32.86
Genetic Algorithm	62s	431	9.4	0.54	0.76	479	32.80

Also included in the table is computation time. The genetic algorithm requires over 30 times the computational effort (62 sec. vs. 2 sec.) as the gradient search method. Given that both algorithms produce valid solutions (the gradient algorithm is actually slightly better by 0.1%), it is reasonable to criticize the greater than 30x time penalty associated with the GA method. However, we must keep in mind the fact that our problem has been simplified to analyze a

[§] Normalized geometric selection is similar to a Roulette Wheel selection, but normalizes the fitness values geometrically.
[**] The standard deviation of the Gaussian distribution is reduced over generations, causing the variable band to narrow from generation to generation. This method is similar to cooling schedule in simulated annealing.

continuous region of the design space by holding the discrete design variables constant. The gradient method simply does not work once the problem is scaled up to include all design variables and the full design space.

GA Tuning

All heuristic methods, including genetic algorithms, require proper tuning to ensure a good balance between design solution quality and computational efficiency. Population size and mutation rate were investigated in more depth during this study. The results of multiple GA runs were analyzed to produce recommended performance settings.

As mentioned previously, there are 4 types of mutation in the GAOT implementation- boundary, uniform, non-uniform and multi-non-uniform. The ratio between the individual mutation rates is 2:2:2:3, respectively. A Mutation Factor (MF) was defined that acts as a multiplier on this basic ratio:

$$Mutation\ array = MF * \begin{bmatrix} 2 & 2 & 2 & 3 \end{bmatrix}^T \qquad (2)$$

The value of MF was varied from 1 to 3. Population size was also varied from 20 to 100 in increments of 20 members. In addition, the termination criterion was set up to be the minimum of 1) 100 generations or 2) within 0.1% of the best gradient result. The second termination criterion was set to the best gradient result to tune the number of generations required without having to perform separate trials. For each population size and mutation factor, five trials were performed to reduce the effects of randomness in the genetic algorithm on the results. Table 11, Table 12, and Table 13 show the average results of the best value for the objective (speed), the computation time, and the number of generations respectively.

The overall best performance in Objective Function value was obtained using a population size of 100 and a mutation factor of 2. The optimum speed was 32.8 knots. While this set of tuning parameters did not have the fewest number of generations, nor the best time, the overall average across all the combinations in consideration of solution, time and number of generations makes it the 'winner'. Mutation factor seems to be the greater driver of performance. While decreasing population size increases the computation time, a Mutation Factor of 2 resulted in near optimum results in all 25 runs. Based on the results from this algorithm, a different mutation factor may not only take a longer time to calculate, but may not reach a near optimum. Furthermore, a larger population size may further increase the performance of the GA, although this was not investigated.

Table 11: Five run average speed

	Mutation Factor (MF)		
Pop Size	1	2	3
20	19.67	**32.84**	32.76
40	26.11	**32.83**	32.79
60	32.80	**32.81**	26.27
80	26.20	**32.84**	32.82
100	**32.80**	**32.83**	**32.84**

Table 12: Five run average computational time

	Mutation Factor (MF)		
Pop Size	1	2	3
20	43.87	52.03	87.07
40	36.09	45.07	62.13
60	32.25	35.17	56.52
80	30.62	32.93	65.47
100	27.12	51.62	37.68

Table 13: Five run average number of generations

	Mutation Factor (MF)		
Pop Size	**1**	**2**	**3**
20	95.4	71.0	85.8
40	86.0	63.4	62.2
60	79.2	53.0	61.6
80	82.2	50.6	70.8
100	74.6	78.6	41.0

Another result from this analysis was the effect of the mutation factor on the calculation time. As mutation factor increases, the computation time also increases. In fact, it has a more significant impact than population size. This is caused by the implementation of the mutations. Mutation is performed on a set number of individuals in this implementation based on the mutation factor. As the mutation factor increases, the number of mutations increases proportionally. Since the algorithm only re-evaluates a particular chromosome if its composition has changed, this directly increases the number of simulation code calls. Therefore, the only difference caused by the population size is during initialization. (All designs must be evaluated in the first generation.) The absolute nature of the mutation factor may not be the best implementation, given that it is not truly a rate when compared between different population sizes. Future work could modify the mutation and crossover rates to be true rates, ensuring a change is made to a set proportion of the population.

Extension of the Genetic Algorithm to Multi-Criteria Analysis

The genetic algorithm implementation described in the previous section was extended in order to evaluate four objectives- speed, range, combat capability, and cost. In keeping with standard practice in ship design, the first three objectives were merged into an overall measure of effectiveness (OMOE). The genetic algorithm was then used to perform a multi-objective optimization of cost versus OMOE.

The GAOT used for single objective optimization did not have an imbedded capability to perform multi-objective optimization. A toolbox called GOSET, developed at Purdue University, was used to perform this optimization. Like the GAOT, it uses a floating point representation of the chromosome. However, it is different in its implementation of the various genetic operators. Selection is done using a roulette wheel approach similar to GAOT. Crossover is performed using a simulated binary method where each variable of the two parents are increased and decreased the same amount on the two children such that the average value for the parents and children remain the same. The amount of movement is based on a probability distribution where shorter movements are more likely. This form of crossover is performed on each variable in the two parents to create two new children. Sixty percent of the population is selected for crossover.

Mutation is done using four operators. Each operator has a small percentage chance of being applied to a variable. The first operator is total gene mutation where there is a 0.1% change of assigning a new value to a single randomly within the variable bounds. The second operator is relative gene mutation where there is a 0.2% chance of multiplying a variable by a fraction determined from a Gaussian distribution. The third operator is absolute gene mutation where there is a 0.2% change of an amount is added to a variable based on a Gaussian distribution. The fourth operator is similar to the first but only operates on integer variables. This operator has a 0.08% chance of occurrence.

Finally, to enable the genetic algorithm to better locate the Pareto front, two additional operators were used: elitism and diversity control. At each generation elitism causes the genetic algorithm to retain the non-dominated designs found up to that point. Up to 50% of the population set can be devoted to these "best" solutions. Diversity control is also used to spread the solutions across the Pareto front. Diversity control works by applying a fitness penalty to a design based on how many designs are in close proximity, ensuring that all designs do not cluster in one area of the Pareto front.

PARETO FRONT ANALYSIS

The four objectives (range, speed, combat capability, and cost) are generally considered to impact design in opposition to one another. For example, for a fixed combat capability, increasing speed also increases cost. Likewise, increasing range decreases speed. These effects are seen clearly in Figure 4 and 5. As we have noted before, the ruggedness of the graphs indicates the significant nonlinearities of the problem.

Experienced decision-makers could perform design selection using information derived from Figures 4 and 5 if current practices generated this type of information. Unfortunately, in general, this type of information is not presented to decision-makers. Furthermore, visualization and data processing becomes a greater challenge as we move beyond bi-objective problems or pair-wise comparisons of objectives. In an effort to make the optimization outputs more manageable, an Overall Measure of Effectiveness (OMOE) was utilized.

OMOE is a linear combination of three "orthogonal" objectives: range, speed and combat capability.

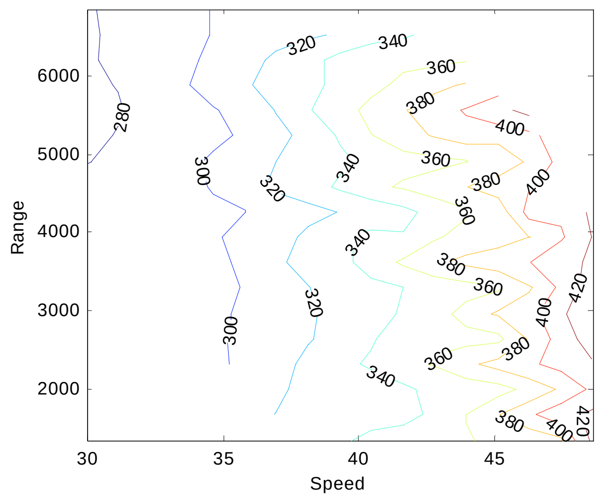


Figure 4. Pareto Front of Speed, Range, and Cost for Designs with Combat Capability at 1.0

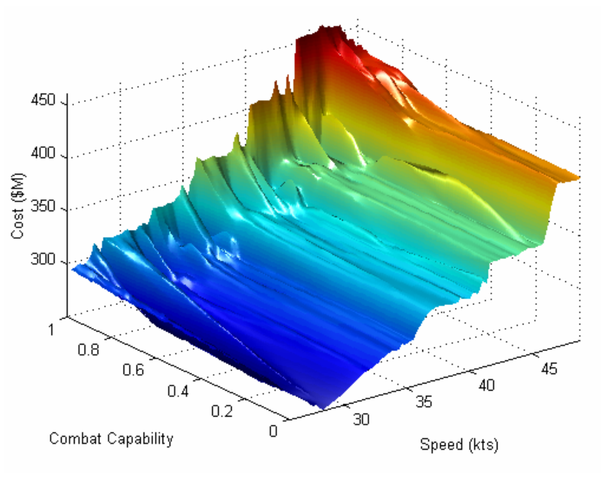


Figure 5. Pareto Front of Speed, Combat Capability, and Cost for designs with Range Over 5000nm

$$OMOE=.15*Range+.3*Speed+.55*CombatCap \quad (3)$$

In order to use the above equation, the results for range and speed were normalized:

$$\begin{aligned}&\textit{Speed}: [20,45]\rightarrow[0,1]\\&\textit{Range}:[1000,5000]\rightarrow[0,1] \qquad (4)\end{aligned}$$

Values outside the above ranges were assigned at the normalized distribution limits. For example, a speed of 15 kt. is assigned a value of 0. Combat capability is reported as a normalized value by the model.

In this case, a Pareto front has been created based on the Overall Measure of Effectiveness (OMOE) and cost. Figure 6 shows the results of two optimization methods used to determine the Pareto Front- a combinatorial mesh (shown in blue) and a genetic algorithm (GA) (shown in red). The non-dominated front, built from the combined data set, is shown in green. While there is no guarantee that the true Pareto front has been found, we can observe that the genetic algorithm performed better than the combinatorial mesh, finding better designs (lower cost, better OMOE) than the gradient based optimization algorithm in most cases. This is because the GA allows design variables to span a greater range of the design space whereas the combinatorial mesh is limited to discrete points within the design space. Also note that the two fronts clearly show multiple concave regions, another indication that our design landscape is rugged (complex).

Given the large range of overall ship design options in terms of OMOE and cost, it is important to understand

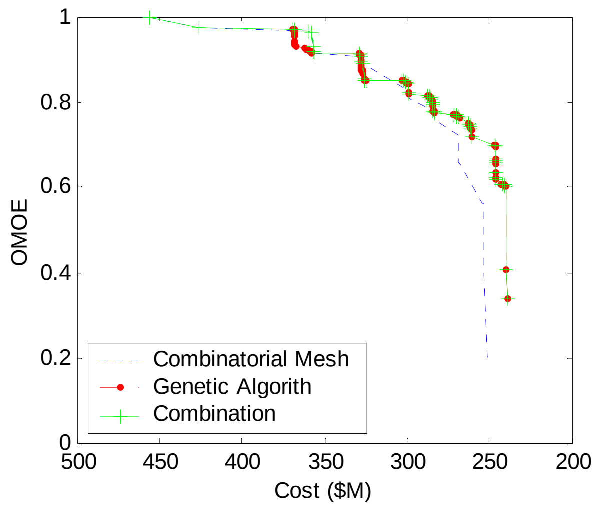


Figure 6. Pareto Front of OMOE vs. Cost Using Two Methods

how the design description, or design vector, changes with changing position along the Pareto front. Are neighboring designs closely related to one another or is the relation between neighboring designs more random? The question is even more significant due to the unusual stair step nature of the Pareto front. Detailed analysis showed "design clusters" along the front, with drastic changes in either cost or OMOE occurring as discrete variables were changed. The discrete variables most responsible for these clusters were propulsion engine type and configuration and combat payload.

Engine types progress from diesel engines through LM2500 gas turbines to LM6000 gas turbines. Within the engine progression, in general the number of shafts goes from 1 to 2, with the exception of a 2 diesel engine/2 shaft combination that fills an intermediate power void. Figure 7 shows the area of the changes in propulsion. While this paper does not allow for description of the individual combat payloads, in general as cost increases along the Pareto front, the combat capability measure also goes up. There is some variation within the design clusters which accounts for some of the smaller gaps in the steps along the front. Overall, both trends make sense: speed and combat capability increase with cost. The conclusion is that the changes progress in a logical order, but through discrete jumps.

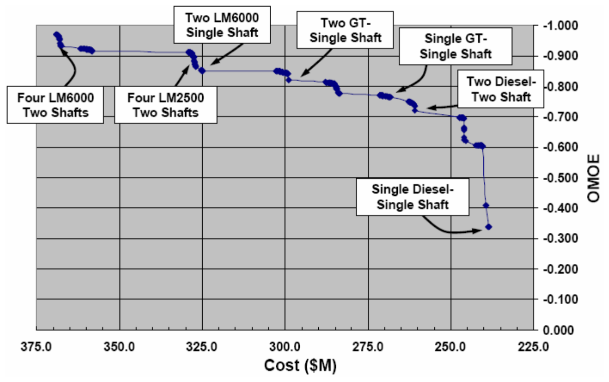


Figure 7: Engine type and shaft number based on cost and MOE.

ANALYSIS OF OVERALL MEASURE OF EFFECTIVENESS

We have shown how the model can be used to develop a Pareto front. However, the mathematical nature of OMOE ensures that Pareto fronts generated with this method are merely one of many. More specifically, designs which are on the Pareto front for one particular OMOE formulation may not be on a Pareto front for a differently weighted OMOE. In the case of ship design, for example, our intuition tells us that varying weights given to speed, range and combat capability in OMOE might significantly change the design choice. For example, pushing the combat capability weighting to zero should push designs with high speed and range and the minimum acceptable combat capability to the Pareto front.

Logical analysis and experience with ship design and acquisition processes tells us that the weightings stakeholders place on OMOE components and their reasons for these choices are often not explicitly stated and vary widely between stakeholders. If a multi-criteria decision is made with weighted objectives, the objective weights are typically assigned based on ‘expert’ opinion, usually before any analysis has been performed.[††] In addition, weightings also change over time as the stakeholders’ opinions are altered by interaction with others, new information, etc. Ideally, a robust design should be selected: one that varies little as OMOE weighting changes. A high degree of sensitivity to small weight changes is a warning sign.

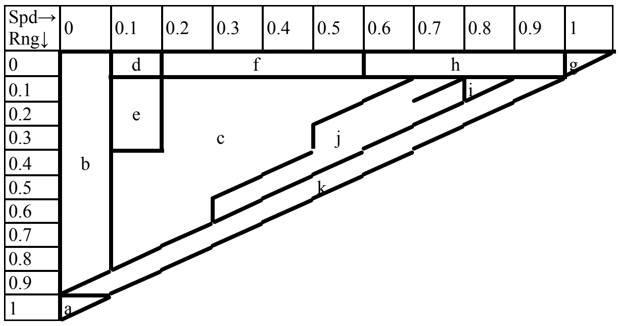
Because of the impact weighting factors could have on design choices, analysis was undertaken to gain insight to how OMOE affects design choices. All factors (speed, range and combat capability) were normalized to produce individual objective function MOEs (as noted earlier). That is, the design range (min, max) for each objective was normalized (to a value from 0-1). Recall that OMOE is a weighted sum of each design objective:

$$OMOE = x*Range + y*Speed + z*CombatCap \quad (5)$$

The weights for each MOE are varied from 0 to 1, with the total weight for all three factors summing to 1. For example, a Speed weighting of 0.7 and Range weighting of 0.2 drives Combat Capability to have a weight of 0.1.

To illustrate how selection of the “best” design varies for each set of weightings, ship cost was constrained to $350M. MOE weights were then varied for all designs below this cost point. (Data from the combinatorial mesh was used for this portion of the analysis.[‡‡]) Design variables were normalized over the following ranges: Speed (Vs) from 20 to 45 knots, Range (Eact) from 1000 to 5000 nautical miles (NM) (arbitrary threshold and goal values for speed and range). The design with the highest OMOE, for the cheapest cost was chosen. In Table 14, each cell contains the optimal design for the combination of Vs and Eact weighting factors. It is notable that only 11 designs (out of 540 non-dominated designs less than $350M) are selected as ‘best’ when the entire range of possible weighting factors is used to calculate OMOE. Our conclusion from this analysis is that design choices are affected by the weighting factors used in the OMOE calculation and that OMOE overlooks many designs that might be preferable to the ones chosen using this method.

Table 14. Design ID vs. Speed and Range Weights



Spd→ Rng↓	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0		d		f				h			g
0.1									i		
0.2		e									
0.3				c		j					
0.4	b										
0.5						k					
0.6											
0.7											
0.8											
0.9											
1	a										

[††] In an ideal case, the weightings would be influenced by an operational analysis based on mission requirements. In practice, it is difficult to say whether this is always the case.

[‡‡] Our GA implementation only considered one set of objective weights in producing OMOE though it is possible to scale the implementation to perform the optimization on any number of objectives.

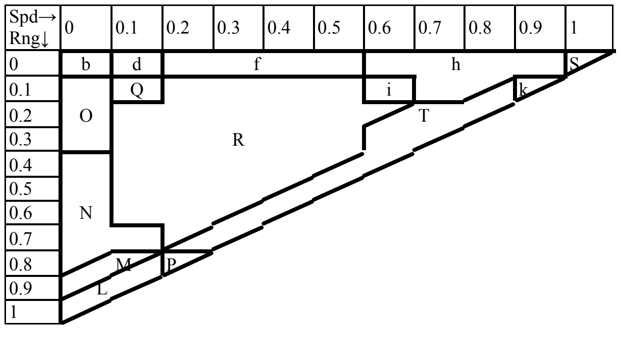
Additional analysis on the impact of individual MOE normalization revealed that this process also significantly impacts the 'optimal' design choice. There are a couple of drivers for this phenomenon. Some designs were eliminated because they were effectively identical when the goal/threshold bounds of the normalization procedure were taken into account. For example, a range performances of 5000 NM and 6000 NM become identical if the goal range is 5000 NM.

To explore this impact, the OMOE analysis was reperformed using the minimum and maximum values of all designs that meet the cost constraint as the normalization limits. For this analysis, the values were:

$$\begin{aligned}&\textit{Speed}\text{: } [25.8,46.0]\rightarrow[0,1]\\ &\textit{Range}\text{:}[1287,7323]\rightarrow[0,1]\end{aligned} \qquad (6)$$

This results in some significant changes to the selected designs as shown in Table 33. Six of the designs from using the goal and threshold limits above were still present, indicated by the lower case letter designation. These designs, however, are at the extremities of the possible OMOE. More significant is the number of designs that change, especially in the center region. The Goal/Threshold value normalization resulted in the design space being dominated by design "c" (Speed 40.7kts, Range 5290nm, Combat capability .95, Cost $329M) while the design space min/max value normalization resulted in the design space being dominated by design "R" (Speed 40.2kts, Range 6732nm, Combat Capability .95, Cost $333M). The impact here is significant: changing the MOE normalization procedure gained an increase of over 30% in range for about 1% less speed and 1% more cost. While it is a matter of opinion which design is indeed better, the fact that a small change in OMOE formulation generated a significantly different capability calls into question OMOE-based design selection methods.

Table 15: Design ID vs. Speed and Range Weights, Without Goal and Threshold Limits.



Spd→ Rng↓
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
b
d
f
h
S
O
Q
i
k
T
R
N
M
P
L

The final issue with OMOE is its linearity. In mathematical terms, OMOE (Eqn. 3), is a plane in 3-dimensional objective-space; changes in OMOE represent a moving parallel plane. Likewise, our Pareto front in objective space can be plotted in three dimensions, representing speed, range, and combat capability (similar to Figure 5). The "best" design is the point where the moving OMOE plane initially intersects this Pareto front. What our OMOE/MOE analysis shows is that changing weights in OMOE changes the intersection point with the Pareto front. The linear OMOE only allows us access to convex regions of the objective space. This may overlook many potentially optimal solutions, especially when the objective space is very non-linear as in the ship design case.

Our conclusions here may be slightly skewed because of the limitations of the combinatorial mesh optimization. Only a few payload options were used and the constraints placed on other variables to run the combinatorial analysis in a reasonable time were also limiting. Performing a multi-objective GA using four objectives- speed, range, combat capability and cost is a next logical step in analysis. However, with more objectives, population size (and computation time) must increase to generate an analytically defensible population of non-dominated designs.

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

The application of optimization tools to preliminary ship design offers potential for more consistent identification of preliminary ship designs that are well-matched with the needs of decision makers. A well-defined design space is effectively explored using automated methods, although radically different ship designs cannot be expected to arise when using an empirically derived model. While it is theoretically possible to do major topological shifts (e.g. being able to move from a mono-hull to a tri-maran) required for truly radical design optimization, the theoretical models and optimization techniques are still being explored. Experts are still needed to conceptualize and define radically innovative design options, barring further advances in artificial intelligence and topological optimization.

This work has demonstrated that the heuristic nature of genetic algorithms is superior to gradient based methods for exploration of large, highly nonlinear design spaces. Genetic algorithms are able to handle the discrete variables that are common when making preliminary architectural choices for all types of designs. Gradient methods, on the other hand, have been demonstrated to be more effective in terms of local exploration of continuous design spaces. Our analysis shows gradient methods to be approximately 30 times faster than a GA when exploring a continuous region, while also producing a slightly superior result. Given this performance advantage, a hybrid approach is recommended. GA's should be used for global space exploration and selection of discrete variables while gradient based methods can be applied in parallel to fine-tune local solutions. This combination of methods in parallel has the promise of being superior to either method alone.

Our work has explicitly exposed two frequently unacknowledged deficiencies in preliminary ship design processes. The root of the problem is the need to match data representation with human cognitive limits. Once the values of each of the three objectives (four, counting cost) have been calculated, how can they be presented in such a way as to allow a human to effectively process the information and then make an informed decision? The need to visualize the outputs of the optimization process leads us to the second limitation of multi-objective analysis- the use of a weighted sum of objectives, OMOE in the case of this effort. OMOE collapses three design objectives (speed, range and combat capability) to a single number. Through the process of creating an OMOE, an artificial reduction in complexity occurs; relationships between the "real" objectives that are significant factors in determining the overall 'optimal' design are hidden from the decision maker.

Also, normalization across objectives is required in order to compose a weighted objective function, such as OMOE. Methods of normalization such as the use of minimum and maximum observed values or goal and threshold values were demonstrated to generate different 'optimal' designs. Min and max observed values have their benefits because they do not depend on "expert opinion," but they may overweight minimal improvements in performance when the range of values is small.

There are two potential solutions to the problem of OMOE. The first involves more care and greater granularity in determining OMOE. Techniques such as multi-attribute utility theory and concurrent design hold promise in this area. Alternatively, rather than relying on our ability to determine the utility of various objectives a priori, followed by design exploration and selection, MDO allows the design exploration phase to determine the multi-objective Pareto front. With the knowledge of the structure of objective space, decision makers can make a more informed decision and be more confindent in the results.

In conclusion, a hybrid combination of genetic algorithms and gradient-based optimization techniques is recommended to augment the preliminary ship design process. The OMOE method for aggregation of design information was proven to have questionable merit and alternative methods should be investigated.

REFERENCES

[1] Wolf, R. *Ship Design Optimization Using MATLAB. Unpublished paper. April, 2004.*

[2] Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning*, Reading, Addison-Wesley Publishing Company, Inc., 1989.

[3] Sudhoff, S.E. *GOSET Manual Version 1.03*, Purdue University, 2004.