Generating Optimal Wing Designs Using Evolutionary Strategy Algorithms

Jacob Baxter[1]
The United States Military Academy, West Point, NY, 10997

Wing Designer is a program which allows students in an introductory aerodynamics course to explore the effect of varying aircraft wing parameters in order to develop knowledge about design tradeoffs. A shortcoming of Wing Designer is the lack of an objective way to rate the students' design choices made from a "large" design space. The purpose of this study is to develop an effective way to generate an optimal set of solutions in order to provide a basis for comparison of design performance and to explore unconventional design possibilities. This study uses a heuristic optimization technique, Covariance Matrix Adaptation Evolutionary Strategy (CMA) to search for an ideal set of solutions. This study modified the Wing Designer user interface from a graphical user interface (GUI) to a text-based input. Also, Wing Designer is adapted with different constraint methods in order to prevent CMA from searching outside of the set of physically plausible solutions. CMA shows a high level of potential ability to generate an ideal set of solutions. The analysis offers insight into the design space of Wing Designer, serving as a useful tool for error checking the design possibilities provided by Wing Designer. Results provide analysis used to improve the robustness of Wing Designer by admitting only feasible designs.

I. Introduction

Evolutionary algorithms are a branch of mathematics and optimization techniques. They are heuristic search algorithms which attempt to optimize inputs within a solution space based upon the performance of inputs that are tried. This research began by evaluating the potential of the different types of evolutionary algorithms for application to optimal wing design, to include genetic algorithms and evolutionary strategy. This study uses a particular formulation of evolutionary strategy, Covariance Matrix Adaptation – Evolutionary Strategy (CMA).

This study uses CMA to optimize the results of a program titled Wing Designer. Wing Designer is a program designed by and used within the aeronautical engineering track at the United States Military Academy. Students within the aeronautical program use Wing Designer to help develop an idea of the design trade-offs within wing design. The process of interfacing CMA and Wing Designer presented many design challenges. Interfacing required consideration of the designer's thought process, as well as the underlying aerodynamics.

The first application of CMA required a conversion of Wing Designer from a graphical user interface to text-based inputs which allow CMA to interact meaningfully with Wing Designer. This change in input methods also presented challenges. The uncontrolled interaction between CMA and Wing Designer required building a set of constraints to guide the search within the acceptable domain of inputs, as well as within the domain of physically plausible solutions. This required an iterative design process which tested multiple constraint formulations.

Through the multiple iterative adaptations of the interaction between CMA and Wing Designer, multiple observations and refinements of Wing Designer were possible. These observations allowed for critical analysis of the functionality and robustness of Wing Designer. CMA has displayed potential for being a very capable tool for

[1] 2nd Year Cadet, Department of Civil and Mechanical Engineering, P.O. Box 0328, West Point, NY – 10997, AIAA Student Member

error analysis of the Wing Designer program and solution space. CMA has also displayed a great deal of potential for providing an optimal wing design.

1 A. Wing Designer

Wing Designer is a program written in Matlab. Wing Designer consists of a graphical user interface which accepts a variety of design inputs from the user. This program passes these inputs to different segments of code in order to determine aerodynamic characteristics of the wing. It also scores the design with an objective function. This score provides the user with a way to objectively rate the success of the design. The score evaluates the performance criteria – range, payload, bending moment, velocity, and span – against accepted values of the C-130 airframe's wing. The Wing Designer problem statement requires students to design against the performance of a C-130.

$$Score = Range + \frac{Payload - Baseline\_Payload}{Cost\_Payload} - \frac{|BendMoment| - Baseline\_Moment}{Cost\_Moment} + \frac{Velocity - Baseline\_Airspeed}{Cost\_Airspeed} - \frac{Span - Baseline\_Span}{Cost\_Span} \quad (1)$$

The C-130 values make up the baseline scores. The primary author of Wing Designer, Major Phillip Root, subjectively chose the cost variables as scoring weights based on his conception of important C-130 traits. The score function rewards a user's performance which exceeds that of the C-130 design with a higher total score. The process of optimizing Wing Designer using the Covariance Matrix Adaptation Evolutionary Strategy attempts to maximize this score function.

In order to maximize this score, this study had to overcome multiple obstacles associated with Wing Designer. Many of the obstacles occurred because of assumptions made about how a human user interacts with the program. Two of the main obstacles derived from these assumptions were the method of interfacing with the program and the constraints on inputs. It is possible to use code to manipulate the graphical user interface; however CMA interacts more efficiently with a text based input. Therefore, this study converted Wing Designer from a GUI-based interface to a text-input interface, so that CMA could iteratively manipulate the inputs efficiently. A typical user with basic aerodynamics knowledge generally makes assumptions that constrain the design to be similar to conventional aircraft. These user-imposed constraints no longer exist when an algorithm attempts to optimize the problem. Specifically, CMA produced unconventional results when not constrained, for example a negative cruise velocity. The addition of constraints for CMA increased the robustness of Wing Designer.

1 B. Covariance Matrix Adaptation Evolutionary Strategy (CMA)

CMA is a heuristic optimization technique, a way to search for an optimal value within the problem space. It does not guarantee convergence to the perfect solution, but provides a very robust method for searching difficult problem spaces for a satisfactory solution. CMA's design makes it an evolutionary strategy. Evolutionary strategy is a subset of evolutionary algorithms, which employ certain tools or operators to allow the algorithm to determine ideal solutions. CMA uses a population of inputs, which consists of multiple vectors within the problem space. CMA determines the fitness of the different members of the population, using the most fit members to direct the search. CMA then mutates the population according to a statistical distribution and recombines the mutated members to form a new population. CMA iterates through this process until a maximum number of evaluations or a satisfactory score is reached. This study uses a version of CMA coded in Matlab, which can call Wing Designer in order to evaluate the fitness of the population.

II. Objectives

This study seeks to develop an applicable solution method for Wing Designer. This method will create a process that can be reproduced to aid the design of different wings based on different airframes, such as a C-130 or a Cessna 172. To meet the associated challenges of meshing Wing Designer and CMA, this study also seeks to develop effective methods of discretization and constraint application. At its lowest level, the study aims to find a highly optimal Wing Designer input. However, given the nature of Wing Designer, the study also aims to constrain the optimization within bounds of physical plausibility.

III. Procedure

To begin the process of optimizing inputs and results of Wing Designer, this study manipulated Wing Designer to accept text inputs from the user or CMA. This involved rearranging some of the input collection function calls and writing a single file in which to type the inputs or pass in a vector from CMA. One significant obstacle of this conversion involved the discretization of NACA profiles. The original method of selection of NACA profiles required the user to use a drop down menu to select a profile, which were encoded as integer values. CMA during its population mutation would mutate the NACA values to non-integers, as it would with all other continuous variables. To account for this, this study forced Wing Designer to floor the CMA inputs inside of Matlab. To account for the range of NACA profiles being a discrete set of integers between one and thirty-eight, Wing Designer adds a high penalty for exceeding the range of NACA profiles.

Another significant barrier to interfacing CMA and Wing Designer is CMA's inherent design to perform minimization. This study converted Wing Designer's scoring function to a minimization problem, by minimizing the difference between the score and a large initial value. By overcoming these two obstacles, this study could begin initial optimization attempts and results evaluation.

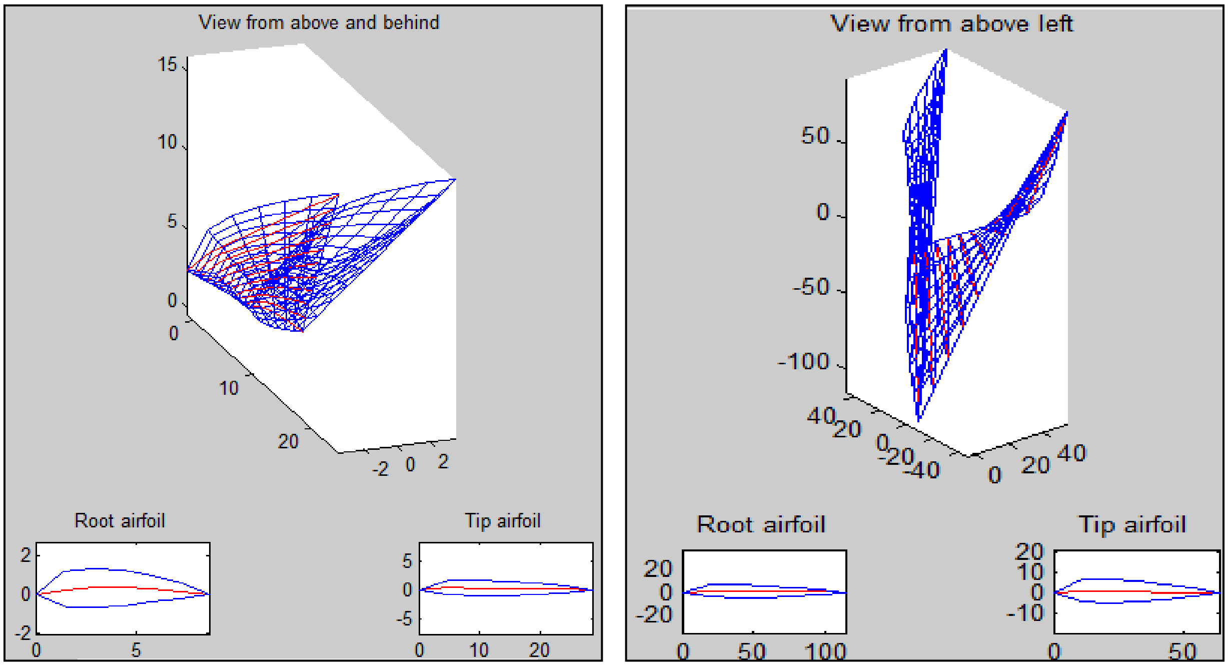
To validate the optimization capabilities of the interface between CMA and Wing Designer, this study used a student solution from a previous year to seed CMA's search. Seeding the solution involved starting the initial search of CMA at this solution. After 20,000 population evaluations CMA had doubled the original score. This initially confirmed CMA's capability to optimize the problem, allowing CMA to begin searching at a random start point. This first attempt used a vector of ones. After 20,000 runs, CMA's search had diverged and the magnitude of the search inputs greatly exceeded useful values. CMA was diverging because the initial start point had been outside of the original constraints set for discretization. In order to circumvent this problem, CMA scaled the ones vector so that the starting inputs were within the constraint boundaries.

With this new start point, CMA generated a wing design that greatly outperformed the improved student solution. However, the wing profile graphically shown did not appear realistic. The wing also consisted of highly unrealistic characteristics, such as a large dihedral greater than thirty degrees. Discussion with a coauthor of Wing Designer, Dr. John Rogers, and an instructor with an aerodynamics background, Major Chris Duling, helped determine and assess the existence of limitations within Wing Designer.[1 & 2] This discussion identified the need to further constrain CMA's search within the "conceptual" limitations that an educated user would impose in their own design process.

This exploration in variable constraints led to the application of a process similar to that used to discretize the NACA profile inputs. After a great deal of tweaking and multiple runs, this method for constraining still caused CMA to diverge, because some boundary conditions could not easily be met by a starting search point. This led to a different method of constraints' application. Wing Designer now applied constraints to these input variables that replaced the score with a high penalty. Wing Designer checked each variable to determine if it was within the proper constraints. If the value exceeded the constraints, Wing Designer calculated a high, additive penalty. Therefore, if more than one variable exceeded the constraints, the penalty increased. The penalty also designed as an exponentially increasing function, based on the value's deviation from the median value within the constraints. This created a penalty which increased exponentially based on the degree to which the input value exceeded the constraints.

CMA was able to find solutions with highly positive scores based on the new constraints, even with random search points that started outside of the constraint boundaries. This led to the identification of additional necessary constraints for Wing Designer and the need for an understanding or metric by which to reasonably apply constraints. A discussion with Major Chris Duling produced a good method for constraining the input. Constraints for different variables would be bounded by current, extreme wing characteristics.[2] For instance, wing span is limited to 300 feet, approximately the length of some production aircraft with very large wingspans.

IV. Results and Discussion



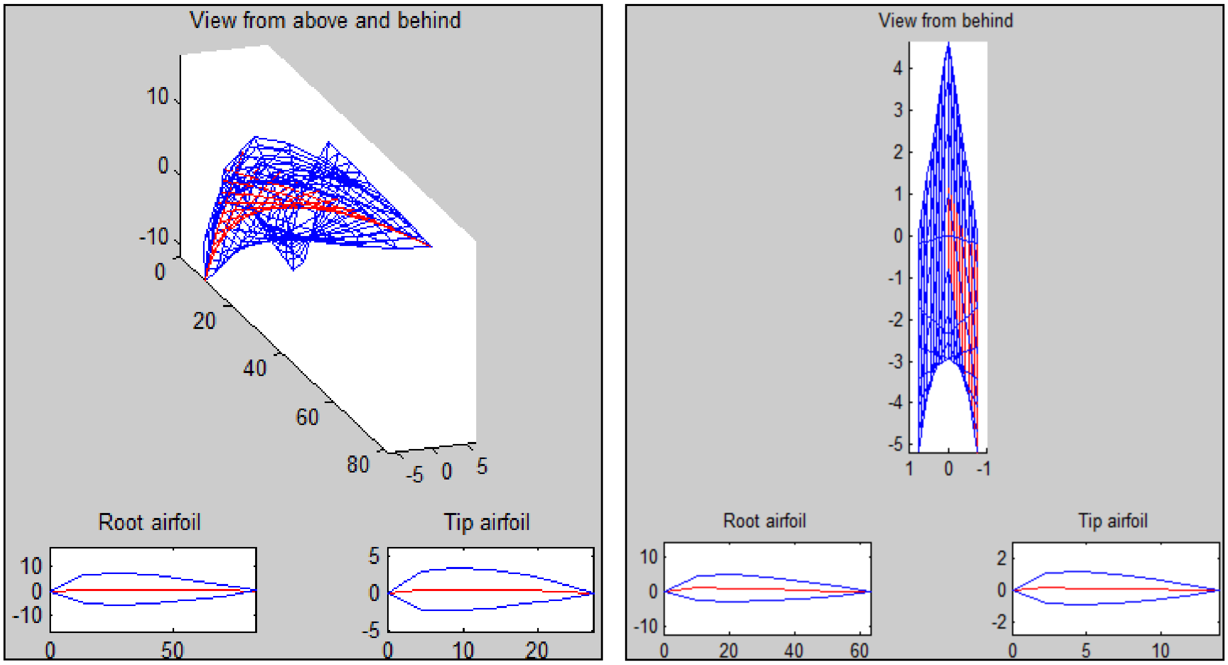
Figures 1 & 2. GUI wing outputs based on initial CMA random search optimization

Figure 1 and figure 2 show initial outputs generated by the GUI, from inputs found by a CMA random optimization search. The study used these initial results in order to draw conclusions about the CMA-Wing Designer interaction. The unusual wing geometries underscored CMA's robustness in exploring the wing design spaces in ways that were unexpected. It also identified certain potential problems within Wing Designer's score calculations.

The wing design in figure 1, also seen in appendix A, has a very small wing span, a negative sweep, a tip chord greater than the root chord, a negative velocity, a negative angle of attack, and a large calculated bending moment. Analysis identified the negative sweep and negative velocity inputs as key implausible inputs for conventional wing design. Large sweep and a large difference between the root angle of incidence and the tip angle of incidence indentified the wing shown in figure 2, also seen in appendix B, as having unrealistic or unconventional inputs. Further analysis showed an unusually large root chord, a large dihedral value, an unrealistically low velocity, and a large calculated bending moment.

The identification of this information led to the development of constraints on negative velocities, negative sweeps, tip chords greater than root chords, very large sweeps, and large differences between the two angles of incidence. Constraint application addressed these issues in small increments. The initial application of constraints allowed the CMA interface with Wing Designer to still function and find solutions.

Analysis of large bending moments led to the examination of the scoring function and the bending moment calculator function within Wing Designer. Evaluation showed that the scoring function was subtracting the bending moment difference from the overall score, as intended. However, if a negative bending moment existed, due to the algebraic design of the scoring function it would add back into the score. This caused a more positive score. Even in this circumstance, a negative bending moment is still poor wing design, because too large of a bending moment in either direction will cause material failure. Therefore, applied modifications to Wing Designer restored the original intent of the scoring function by taking the absolute value of the bending moment, in order to penalize for a large moment in either direction.



Figures 3 & 4. GUI wing outputs based on refined CMA random search optimization

Figure 3 and figure 4 show further wing designs generated by CMA input fed into the GUI. They accurately display small constraints of the input variables, eliminating negative sweeps and negative velocities. Figure 3 and 4 also remphasize earlier unconventional aerodynamic qualities uncovered by the search. The wing displayed in figure 3, also seen in appendix C, shows a very complex and hard to discern geometry. The wing displayed in figure 4, also seen in appendix D, shows a kite-like design with an extremly small span of appoximately 1 foot.

Figure 3 identified an average chord larger than the span, a trait also seen in figures 2 and 4. Figure 3 also consisted of a small span, a large, negative dihedral, a large sweep value, and a negative angle of attack. Figure 4 aso consisted of a large root chord, an extremely small span, and a low velocity. Discussion with Major Chris Duling and Major Phillip Root confirmed the aerodynamic shortcomings of a low velocity, concluding that it would lead to stalling.[2 & 3] Analysis identified the existance of a breakdown in Wing Designer’s scoring capability with large dihedral values, which correlated with separate analysis done by Major Jame Bluman, course director for Introduction to Aerodynamics at the United States Military Academy.[4]

This analysis led to the application of additional constraints using the same constraint approach that limited low velocities, large dihedral values, and encouraged reasonable ranges for chord width. However, the ranges of these additional constraints began to exceed indiscriminate seeding points for the search functions. For instance, a constraint applied on the velocity requiring it to be above a speed of 90 knots, would require the application of a non-random scaling vector to insure its value was within a scoreable domain. Figure 5 represents the constraint method initially applied in order to limit the input values from exceeding their desired space. A comparison between a starting velocity value outside of the constraints and a starting value for wing span outside of the constraints helps understand this situation.

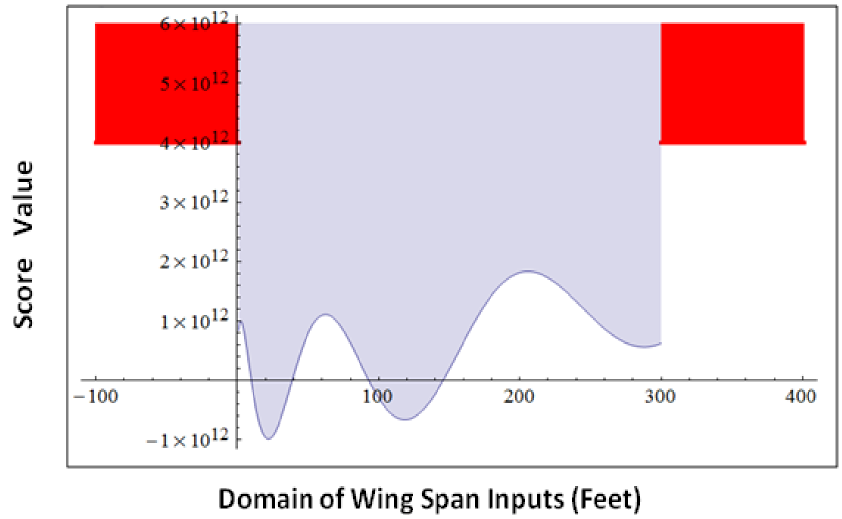


Figure 5. Model of original constraint application to input for wing span

The red area of figure 5 represents the range in which the penalty is applied. The blue area represents a possible domain representation of the aerodynamic score. As soon as a value for wing span exceeds 1 or 300, it is immediately penalized to a high value of 4 x 10^{12}. As CMA iterates through its refining of the search, it gains no sense of direction from such an application of constraints. Its scaling factor for search increases in size. The search value for wing span begins to greatly increase, and only a very lucky guess could be within the non-penalized domain yielding a potentially minimal search score, which would maximize the score within Wing Designer.

This method of constraints application proved to be acceptable for use with a limited number of constraints, approximately two to three, not including the constraints on the discretization of the NACA profiles. However, the application of multiple constraints with some initial values in the penalty region led to an unproductive, divergent CMA search.

Consideration of the lack of direction and useful, efficient ways to constrain the search led to the development of an exponentially, additive penalty, modeled by Eq. (2),

$$Penalty\ Score = (x_a - a_0)^2 + (x_b - b_0)^2 + \cdots + (x_n - n_0)^2 \qquad (2)$$

where the penalty function is only applied if a logical evaluation determined the value is outside of the constraints. Figure 6 displays an example penalty application to wing span.

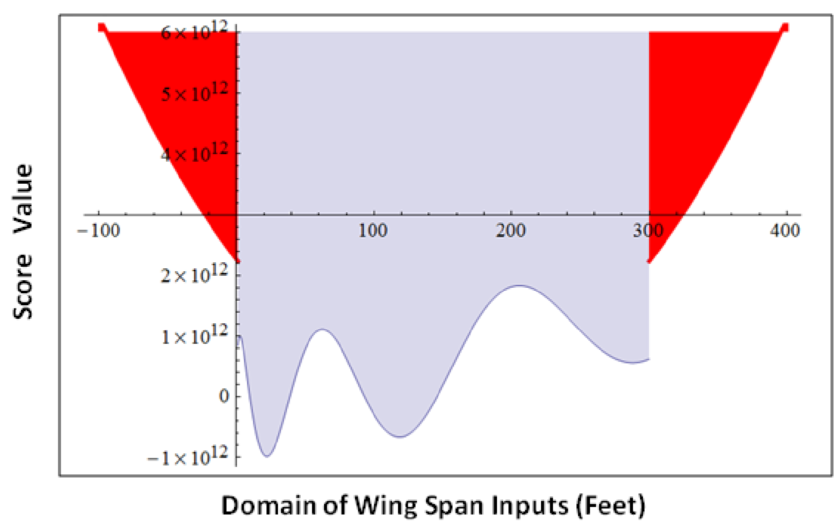


Figure 6. Model of new constraint application to input for wing span

Comparison between figures 5 and 6 shows the added direction given through the new constraint method. The additive nature of the entire penalty score greatly improved CMA's ability to find direction and find its way back into the acceptable problem space.

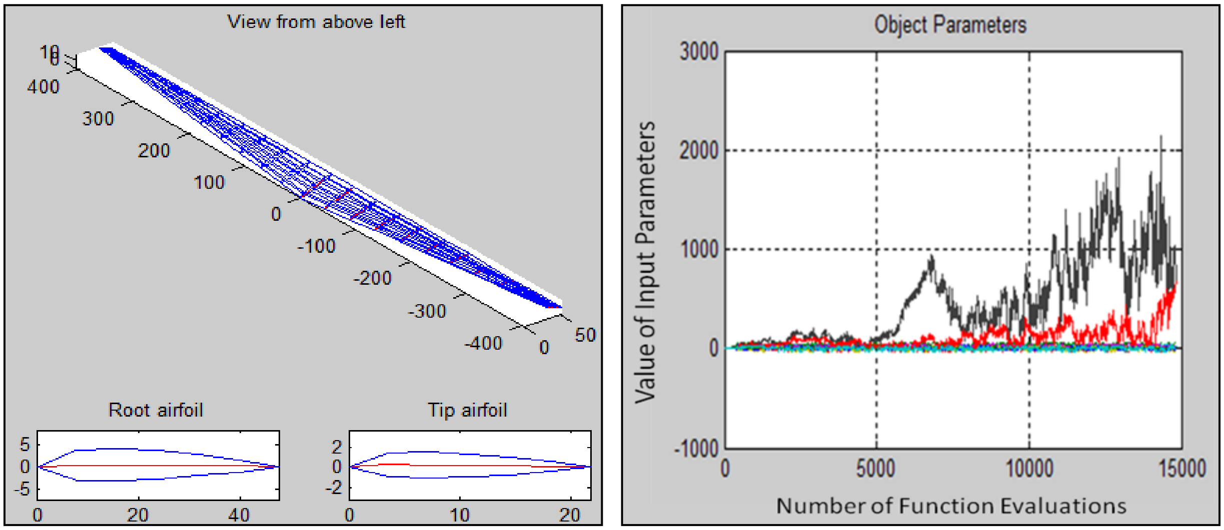


Figure 7 & 8. GUI wing and search output based on refined constraints method

Figure 7 shows a wing design driven by the new constraints. Figure 8 shows the CMA search with the number of function evaluations on the x-axis and the value of the input parameters on the y-axis. The wing in figure 7 and appendix E exhibits many ideal visual characteristics of a conventional wing. It has a gradual taper in chord from root to tip, a long span in comparison to average chord, and a low geometric twist. The only unconventional values are the wing span, which is extremely large at 800 feet, and velocity, which is also large at 690 knots.

The other unconventional and likely physically implausible values have been removed by the constraints handling. Figure 8 displays a concentration of object parameters within the rough order of 10^2, the range required by the applied constraints. The variation of wing span, the black scatter plot, shows CMA's continual search, evaluation, and experimentation with the boundaries of the problem and constraints imposed.

Figures 7 and 8 combined underscore the potential for CMA's application to find an ideal wing design. Further analysis and design conducted have helped reach the following table of constraints:

	Variable Name	Characteristic	Boundary Conditions	Units
1	xmean(1)	**Root** Airfoil NACA Digit	1 < x < 38	**Digit**
2	xmean(2)	**Root** Chord	1 < x < 60 , x < xmean(5)	**Feet**
3	xmean(3)	**Root** Angle	-30 < x < 30	**Degrees**
4	xmean(4)	**Tip** Airfoil NACA Digit	1 < x < 38	**Digit**
5	xmean(5)	**Tip** Chord	1 < x < 60	**Feet**
6	xmean(6)	**Tip** Angle	-30 < x < 30	**Degrees**
7	xmean(7)	**Wing** Span	1 < x < 300	**Feet**
8	xmean(8)	**Wing** Dihedral	-20 < x < 20	**Degrees**
9	xmean(9)	**Wing** Sweep	0 < x < 15	**Degrees**
10	xmean(10)	**Cruise** Velocity	80 < x < 550	**Knots**
11	xmean(11)	**Wing** Angle of Attack	-5 < x < 30	**Degrees**
12	-----	Aspect Ratio	To be determined.	**Unitless**

Table 1. Current constraints applied to CMA search

This table of constraints shows the current search values applied to CMA search. These values create a useful set of limitations on the search. They also provide a possible set of constraints that can be developed into a smart scoring system for Wing Designer. These constraints would allow Wing Designer to develop an additive penalty system that provides more direction for students attempting to learn design tradeoffs. They would also allow any “holes” within the robustness of the applied aerodynamic calculations to be sealed, so that they are not manipulated to provide incorrect information.

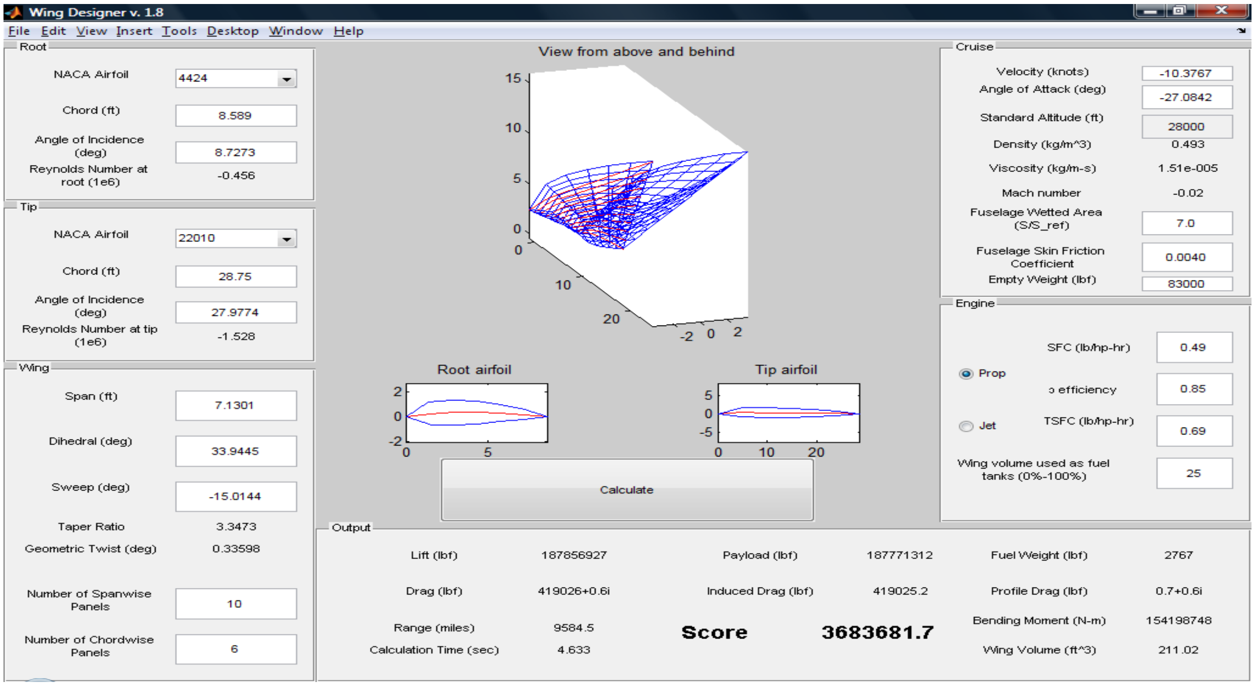
V. Conclusions

Through the iterated process of refining CMA’s search, CMA has demonstrated its adaptability and high performance as a heuristic problem solver. While the scope of this study did not achieve an optimal set of solutions, the methods and development of the techniques and CMA-Wing Designer interface offers a promising method to generate ideal solutions. Using CMA to continue to search the problem space, future work will eventually constrain Wing Designer precisely enough that CMA should demonstrate a relative global maxima. Underlying results point to aspect ratio as one of the final constraints that future work needs to address. The potential solutions generated will offer a new way for instructors of Introduction to Aerodynamics at the United States Military Academy to objectively rate students’ scores based on the optimal existing solutions. This solution method will also offer a potential new approach to consider designing wings, provided the robustness of the underlying aerodynamic calculations of Wing Designer can be verified.

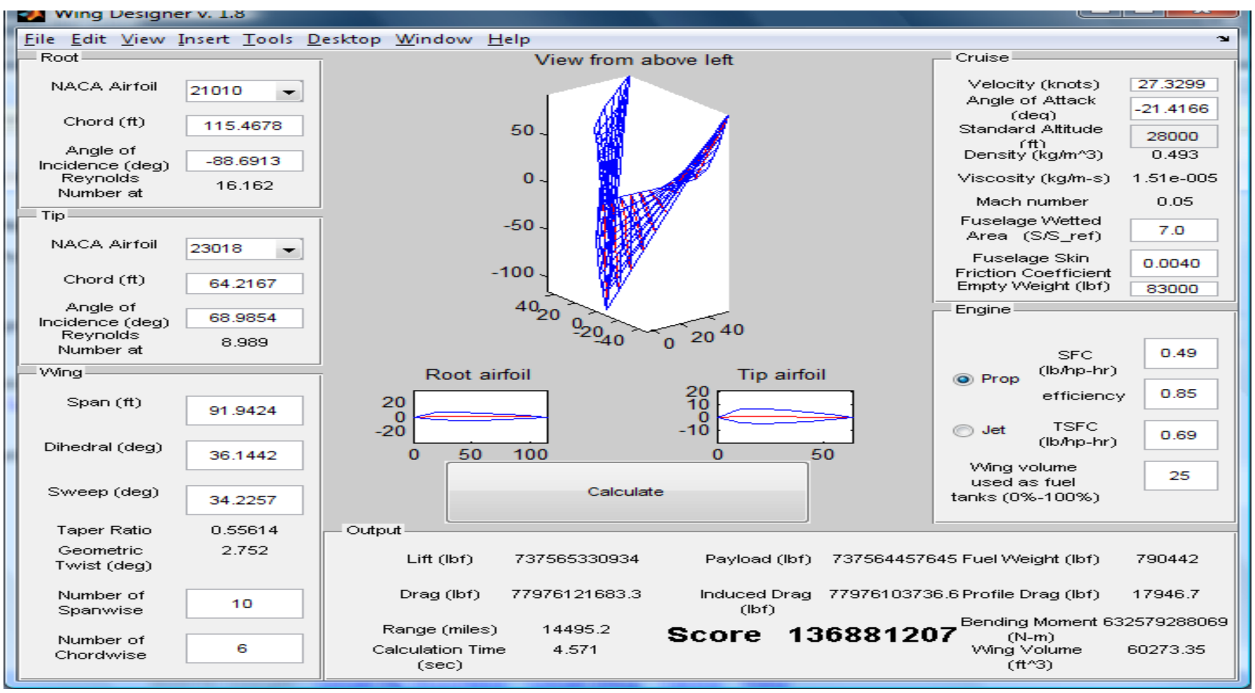
The development of constraints for Wing Designer to guide CMA provides a replicable process which ought to aid in future uses of CMA for other aerodynamic optimization problems, as well as with general engineering problems. The specific boundaries used for constraining Wing Designer provide a novel approach to creating a scoring function which has more adaptive penalties that may provide a more objective and useful score for students. Such an objective score will allow the students to understand and place more emphasis on considering the consequences of their design and the resultant tradeoffs. Between the new scoring method and the identification of potential “holes” in the aerodynamics’ robustness, CMA has helped develop a method for increasing the overall robustness of Wing Designer. This increase in robustness will make it a more useful teaching tool, increasing the potential range of users and students of other aerodynamics’ courses.

There are other considerations for analysis of work done and future work. For the current state of this study, examination into other optimization techniques would be useful for assessing the value of CMA versus other solution methods. Other programs aside from Wing Designer could be used to confirm aerodynamic properties. Student solutions could be reapplied to the new constraints method to insure proper function. Development of a constraint of aspect ratio could provide the final constraint necessary to develop physically plausible wings.

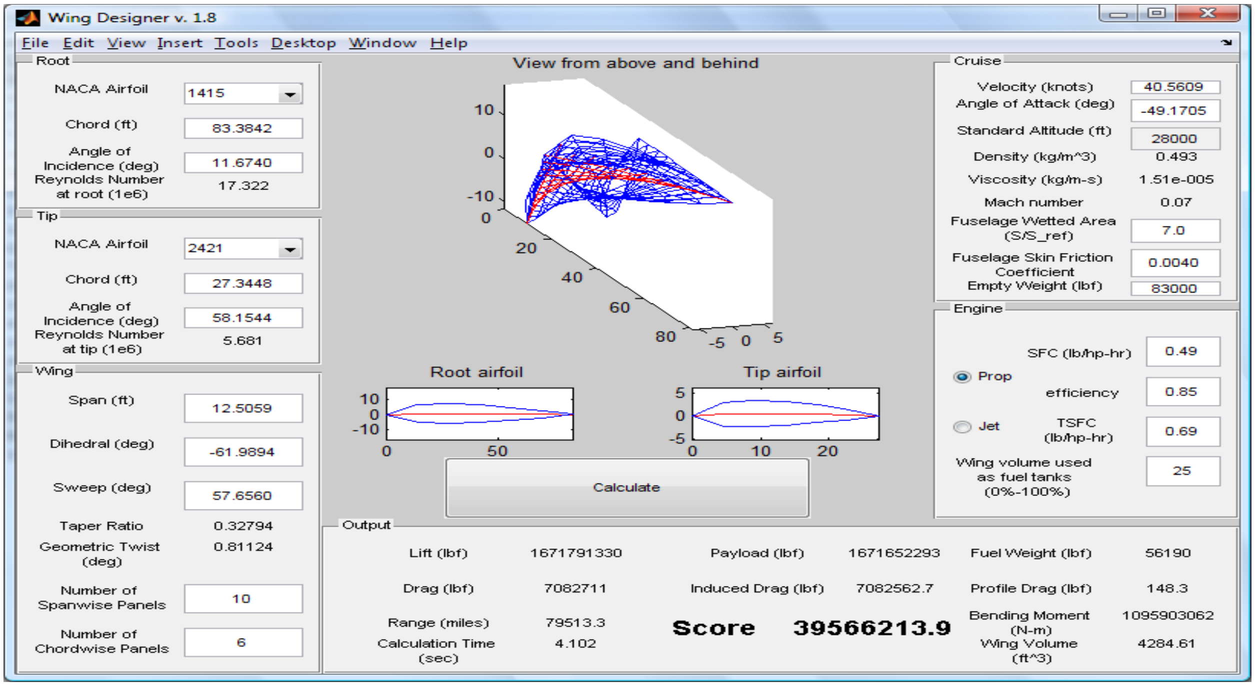
Appendix A. GUI program with inputs and outputs



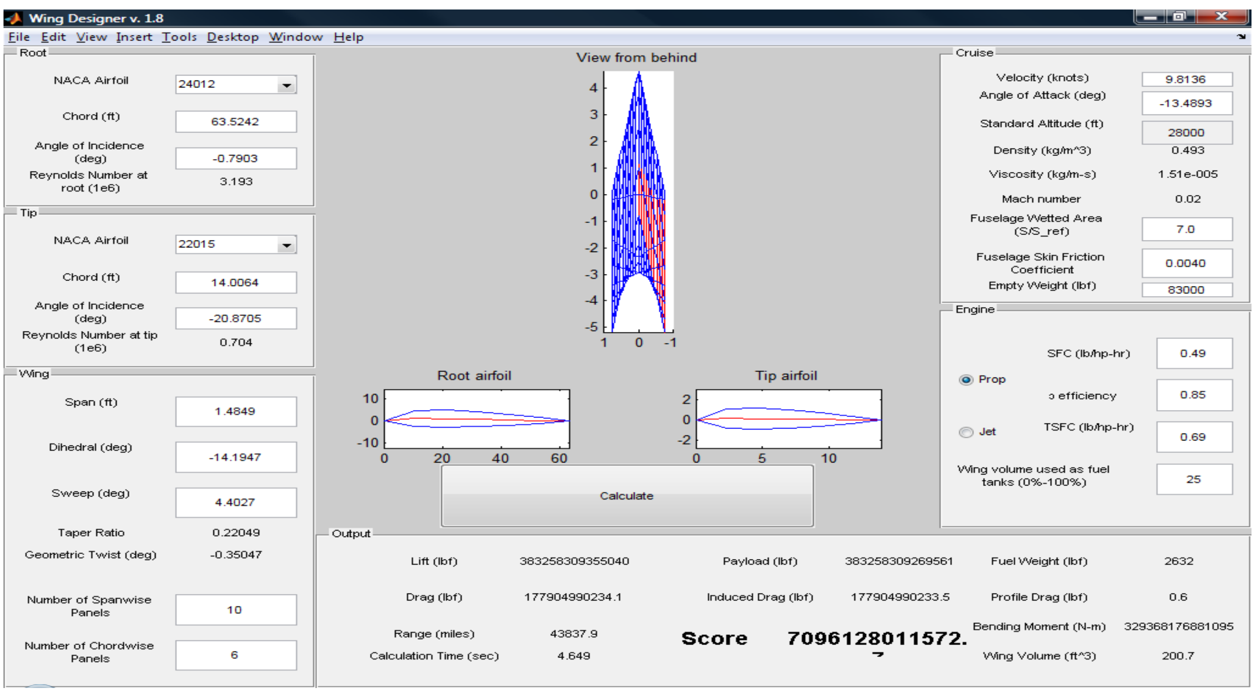
Appendix B. GUI program with inputs and outputs



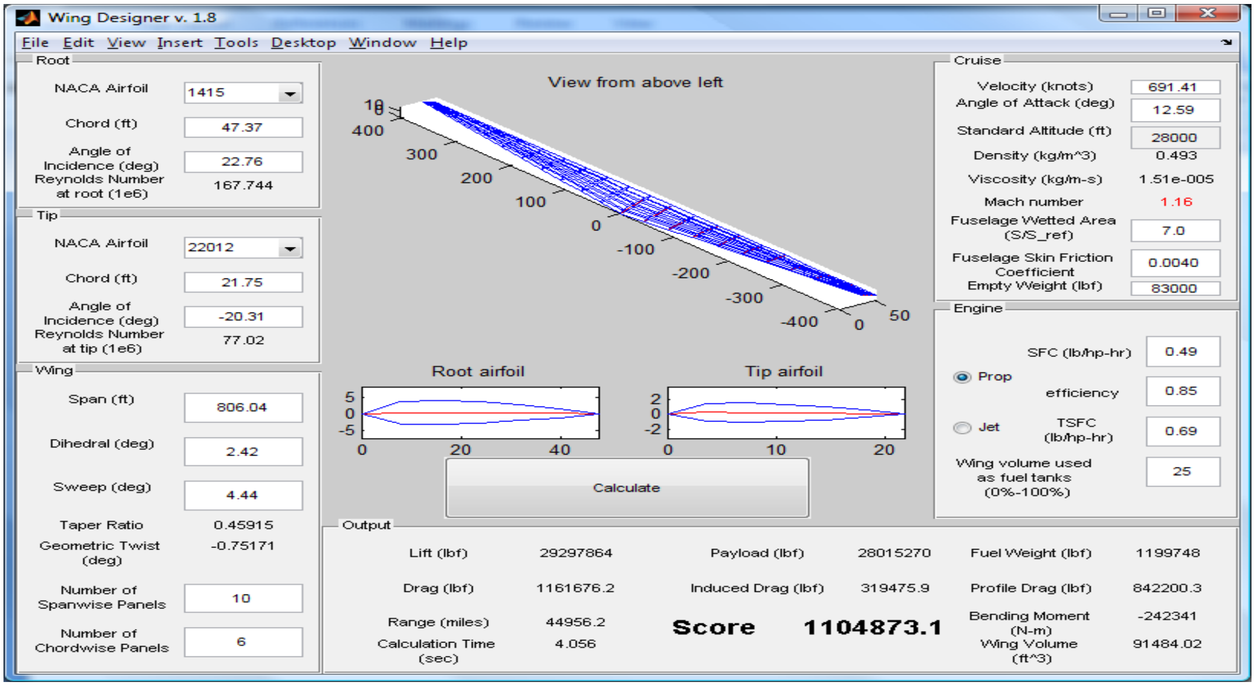
Appendix C. GUI program with inputs and outputs



Appendix D. GUI program with inputs and outputs



Appendix E. GUI program with inputs and outputs



Acknowledgments

I would like to thank Nathan Mayo for his patience and determination in helping me critically proofread my paper.

I would like to thank Majors Chris Duling, Phillip Root, James Bluman, and Dr. John Rogers for their expert and helpful advice, as well as encouragement in helping me reach my goals for this study.

I would like to thank Lt. Col. Donovan Phillips for support, guidance, and direction, acting as a sounding board for many of my ideas and problems.

References

Discussions
[1]Dr. John Rogers. Faculty, Department of Civil and Mechanical Engineering, USMA. Personal Interviews. January to April 2010.

[2]MAJ Chris Duling. Faculty, Department of Civil and Mechanical Engineering, USMA. Personal Interviews. January to April 2010.

[3]MAJ Phillip Root. Former Faculty, Department of Civil and Mechanical Engineering, USMA. Personal Interviews. February to April 2010.

[4]MAJ James Bluman. Faculty, Department of Civil and Mechanical Engineering, USMA. Personal Interviews. March 2010.