Response Surface Methodologies for Low-Boom Supersonic Aircraft Design using Equivalent Area Distributions

Trent Lukaczyk*, Francisco Palacios†, and Juan J. Alonso‡
Stanford University, Stanford CA 94305, USA

The use of response surface models (also known as surrogate models) to describe the performance of aerospace systems within a given design space is by now fairly well established in the MDO community. Despite such widespread use, surrogate-based optimization techniques can incur significant computational expense when high-dimensional spaces must be modeled. In this paper we describe an adaptive, constrained optimization methodology that is based on (a) Gaussian Process Regression (GPR), (b) the use of gradient information, inexpensively obtained via adjoint methods throughout the design space, (c) an adaptive sampling technique based on the notion of expected improvement, and (d) the treatment of nonlinear constraints (within the adaptive sampling loop) so that the number of unnecessary infeasible samples is significantly reduced. The methodology is demonstrated using a number of analytic examples as well as with CFD-based low-boom airfoil and aircraft design problems using constraints based on target equivalent area distributions. This work is part of a larger effort (jointly with Lockheed Martin) to develop a multidisciplinary analysis and optimization framework for the high-fidelity design of low-boom supersonic aircraft within the NASA N+2 Supersonics effort.

I. Introduction

Under NASA's Supersonics Project, Stanford University is collaborating with the Lockheed Martin Corporation to advance the state of the art in design tools for supersonic, low-boom passenger aircraft. Stanford University contributes to this effort by developing a generic Multi-Disciplinary Analysis and Optimization (MDAO) framework capable of identifying aircraft designs that demonstrate increased fuel efficiency by reducing drag, and reduced sonic boom loudness through the use of techniques for optimal shape design.

The present work is focused on the shape design of an N+2 supersonic passenger jet configuration. The latest N+2 program requirements specify that the aircraft must fly at Mach 1.6-1.8 with a minimum range of 4000 nmi and a capacity of approximately 35-70 passengers. In addition, there is a (revised) perceived loudness level target (for ground sonic boom) of 85 PldB, an airport noise target of 10 EPNdB (cumulative) below Stage 4, cruise NOx emissions lower than 10 g NOx / kg of fuel burned, and an efficiency target of more than 3 pax-mi / lb of fuel burned.[1] Our contributions are meant to help directly with the range and sonic boom requirements, and indirectly with cruise emissions and the fuel efficiency target.

We are approaching this problem in two different ways. Firstly, we have developed an adjoint-based, inverse design capability for multiple azimuthal equivalent area distributions generated by the aircraft. In this approach (extended from Ref.[2]) we use high-fidelity CFD and a new adjoint formulation, together with a constrained gradient optimizer to obtain optimal designs whose equivalent area distributions do not depart significantly from a series of pre-specified targets. These analysis and design capabilities build upon our previous work in the field.[1,2] Secondly, the adjoint capability is leveraged to generate gradient information for the surrogate based constrained optimization method we develop in this paper. Based on a stochastic modeling technique known as Gaussian Process Regression, the method includes new approaches to hyperparameter selection and adaptive refinement to enhance the robustness and efficiency of the surrogate model.

*Graduate Student, Department of Aeronautics and Astronautics, AIAA Member
†Engineering Research Associate, Department of Aeronautics and Astronautics, AIAA Member
‡Associate Professor, Department of Aeronautics and Astronautics, AIAA Senior Member

A. Background: Low-Boom Design

In general, to analyze both the performance (C_L, C_D, C_M, etc) and the boom loudness, we perform high-fidelity CFD analyses all the way to a near-field location (typically between 2 and 3 body lengths beneath the aircraft) and extract the pressure signature at a number of azimuthal locations (see Figure 1). Based on classical linearized supersonic aerodynamics theory,[3] we convert these signatures into equivalent area distributions that can then be compared to target distributions known to have good loudness characteristics.[1] In order to carry out design / optimization exercises, the shape of the aircraft is parameterized using arbitrary Free Form Deformation (FFD) control boxes so that we are able to alter the aircraft configuration to minimize drag (while maintaining lift and ensuring that the pitching moment is no worse than that we started with) and ensure that the aircraft equivalent area distributions do not deviate from the target by more than a permitted amount.

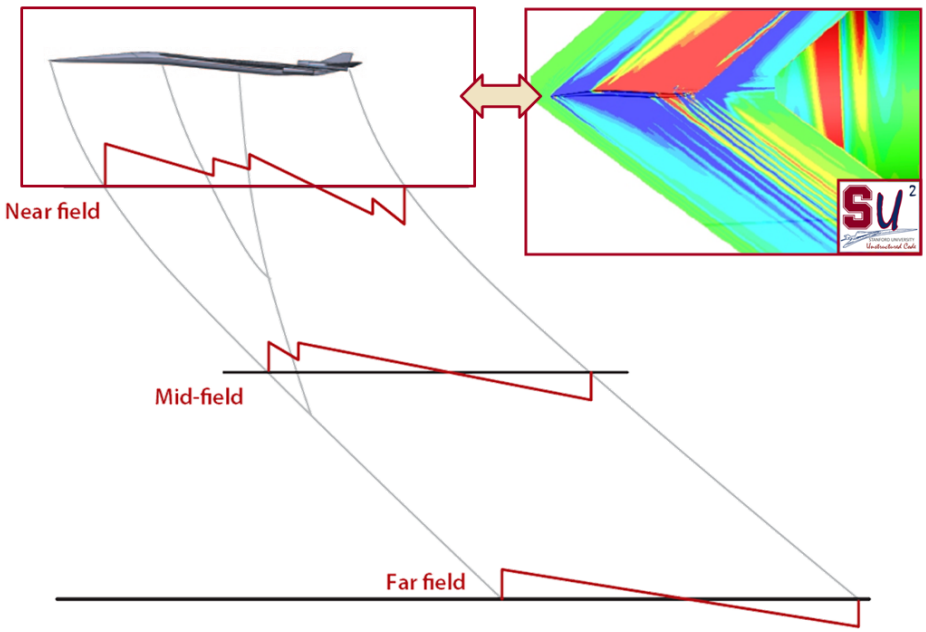


Figure 1: Schematic of CFD-based Supersonic Boom Propagation Simulations

One of the main reasons why we have pursued this inverse-design approach is to attempt to obtain a design space that is hopefully free of discontinuities and that exhibits a smooth behavior. In contrast, design approaches that are based directly on the minimization of sonic boom loudness measures on the ground can lead to difficult design spaces with multiple local minima and sharp discontinuities as was shown by Chung et al.[4] Additionally, the equivalent area analysis for low boom has a well established history and design intuition for supersonic aircraft, and has been used extensively by Lockheed Martin to shape this design to generate the favorable pressure signature it has today. Thus by using the equivalent area inverse-design approach we are hoping to maintain this feature while improving performance.

For all our work, we use the open-source solver SU^2 developed in the Aerospace Design Laboratory at Stanford University. SU^2 is a general purpose partial differential equation solver equipped with tools for optimal shape design including flow and adjoint solvers, free-form mesh deformation, goal-oriented adaptive mesh refinement, and a constrained optimization environment. These tools are wrapped in the Python language to efficiently manage the input and output of data and the exchange of information between the different modules in the SU^2 suite.

In this paper, instead of using the output of SU^2 analyses (function values and adjoint gradients) directly as an input to a gradient-based optimizer, we use the predictions and their gradients with respect to user-specified design parameters to create surrogates models at the lowest possible computational cost. These surrogate models are adaptively improved to guide optimization efforts in a way that minimizes the overall computational cost required to complete an optimization run.

B. Background: Surrogate Based Optimization

To address the high computational cost of the CFD simulations and to enable global design space optimization, we rely on Surrogate Based Optimization (SBO) with gradient-enhanced Gaussian Process Regression (GPR). A surrogate model (also known as a response surface model or RSM) is an N-dimensional surface that characterizes the relationship of design variable inputs and performance parameter outputs by fitting an inexpensive approximation to a set of sample points known as training data. Many MDAO applications have successfully used parametric models (such as Polynomial Regression[5]) and stochastic models (such as Kriging[6]) to generate these surrogates. In the case of SU^2 computations, where we can inexpensively obtain gradient information through the use of an adjoint solution, we attempt to increase the accuracy of the RSM with lower computational cost (see, for example[4, 7–9]). A large amount of work in the past has involved the development and use of RSM techniques, especially (lately) using a stochastic modeling technique known as Kriging[5, 10] and its gradient enhanced relative known as Co-Kriging.[4, 11]

In our work we use GPR, which can be described as super-set of Kriging. GPR approaches regression from a Bayesian standpoint by conditioning a probabilistic function representation to observed training data[12] from a number of runs of a simulation code. It can be shown that, in the case where the probabilistic prior is assumed to be a Gaussian process with a stationary zero mean, the resulting model matches that of Simple Kriging (SK). If the mean of the Gaussian process is assumed to be a combination of a set of basis functions, the resulting model matches that of Kriging with a Trend (KT).[10, 12, 13] If only a constant explicit basis function is used, GPR takes the form of Ordinary Kriging (OK).[10, 12] Because GPR is posed as a conditioning problem and not an expected error minimization problem, it can have more flexibility when handling poorly behaved design spaces (such as discontinuities as found in[4]) by applying non-stationary covariance functions.[14] While this paper will not explore these extensions, they are a key motivation for our current and future exploration of Gaussian Process Regression. Furthermore, while gradient-enhanced GPR has been demonstrated in the context of nonlinear control,[9] and introduced to the design of circulation controlled wings,[15] to the best of our knowledge it has not seen application in supersonic aircraft MDAO problems.

Implementations of Kriging-based surrogate modeling for aircraft MDAO problems have been explored in various works. Chung[4] implements Co-Kriging (gradient enhanced Kriging) with hyperparameter selection by marginal likelihood estimation for the minimization of drag and boom noise at constant lift for a supersonic business jet. Wintzer[16] constructs Kriging response surfaces for the multifidelity design optimization and mission analysis of an oblique wing supersonic jet. Ordaz[17] presents a framework that uses Kriging surrogate modeling tools available within Model Center's Design Explorer software to minimize boom noise for a supersonic aircraft geometry built with Vehicle Sketch Pad.[18]

There are several challenges to performing GPR-based surrogate based optimization. A primary challenge is that a set of free (or hyper) parameters must be chosen with a secondary optimization step. These parameters can be difficult to control and can result in undesirable behavior of the response surface.[19] Second, the learning techniques for these hyper-parameters can become expensive with large numbers of parameters and training data.[19, 20] Third, the response surface can become numerically unstable or ill-conditioned when training data are sampled very close to each other within the design space.[21] Fourth, numerical simulations of complex geometries and physics may produce inaccurate or noisy response data.[22] Finally, refining the response surface requires the solution of another secondary optimization problem, which can become difficult to solve as the response surface converges on the local or global minimum.[20]

II. Methodology: Surrogate Based Optimization

In this section we will present a Gaussian Process Regression framework for surrogate based optimization with enhancements for accuracy, robustness and efficiency. First we will first outline the development of GPR. This will include a brief derivation from first principles, as well as a description of an underlying covariance matrix and the secondary optimization problem required to tune it. We will also comment on including gradient information for accuracy and noise models for stability. Second, we will outline the overall surrogate based optimization method we have developed, including considerations made for increasing the robustness of the hyperparameter learning step, and increasing the efficiency of adaptive sampling criteria.

A. Introduction to Gaussian Process Regression

1. GPR Mathematical Description

Following the derivation given by Rasmussen,[12] Gaussian Process Regression is approached by conditioning a multivariate normal distribution

$$f \sim \mathcal{N}\left(\mu, [\sigma]\right), \tag{1}$$

where f is a normally distributed function with mean vector μ and standard deviation matrix $[\sigma]$. For this paper, we take a uniformly zero mean vector, and populate the standard deviation matrix with a covariance submatrix k that is a function of training and estimated data:

$$\begin{gathered}\begin{bmatrix} f_p \\ f_k^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} k(x_p, x_q) & k(x_p, x_j^*) \\ k(x_k^*, x_q) & k(x_k^*, x_j^*) \end{bmatrix}\right), \\ \{\, f_i(x_i) \,|\, i = 1, ..., n \,\} \;,\; \{\, f_t^*(x_t^*) \,|\, t = 1, ..., m \,\}.\end{gathered} \tag{2}$$

The notation $(\cdot)^*$ is used to distinguish the estimated data from the training data. Additionally, index notation is used to describe the sub-blocks of the covariance matrix, where $k(x_p, x_q)$ would be equivalent to the matrix k_{pq}. There are n training point vectors x with function values $f_i(x_i)$, and m estimated data point vectors x_t^* with function values $f_t^*(x_t^*)$.

In this complete set of data, we are trying to obtain estimated function values f^*. These values are obtained through GPR regression using the training data locations x, the associated function values $f(x)$ that have been computed, as well as the desired estimated data locations x^*. Following Rasmussen's derivation,[12] we condition the normal distribution with the data we do know

$$f|x^*, x, f \sim \mathcal{N}\left(f^*, \mathbb{V}[f^*]\right), \tag{3}$$

which allows us to identify useful relations for estimating a function fit,

$$\begin{gathered}f_k^* = k(x_k^*, x_q)\, k(x_p, x_q)^{-1}\, f_p \\ \mathbb{V}[f_k^*] = \left(k(x_k^*, x_j^*) \;-\; k(x_k^*, x_q)\, k(x_p, x_q)^{-1}\, k(x_p, x_j^*)\right)_k\end{gathered} \tag{4}$$

where $\mathbb{V}[f^*]$ is the covariance of the estimated value f^*. These are the relations needed for coding a GPR program. Rasmussen provides an example algorithm that simplifies these relations by using Cholesky decomposition.[12]

2. Covariance Function

The covariance (or kernel) function models the spatial correlation between data points. It is chosen based on the types of functions that are going to be modeled. Highly-smooth or weakly-smooth functions would be examples of different types of functions that would require different choices of covariance functions. A common covariance function is the Gaussian function of the Euclidean distance between points:[12, 15, 20, 23]

$$\begin{gathered}k(x_p, x_q) = k(p, q) = \theta_1^2 \exp\left(-\frac{1}{2\theta_2^2}\sum_{z=1}^{d}(p_z - q_z)^2\right) \\ \{p_i, q_i, \tfrac{\partial}{\partial x_i} \mid i = 1, ..., d\},\end{gathered} \tag{5}$$

where d is the number of dimensions, and p and q are the position vectors chosen from the design space x. There are two degrees of freedom in the covariance function. These are known as hyperparameters. In terms of their effect on the function fit, the nominal variance θ_1 is a measure of the amount of variance allowable between points, and the length scale θ_2 is a measure of the range of influence of a point.

The choice of covariance function is not arbitrary. Only those that generate a positive definite covariance matrix are permissible as this ensures that the matrix is theoretically invertible. However, it is possible to make this matrix numerically singular if two training points are sufficiently close together, depending on the choice of hyperparameters, because this results in two nearly identical eigenvalues. The Gaussian kernel is known to be especially susceptible to becoming numerically singular because of its flat parabolic shape near

the kernel center. In our experience adding gradient information quickly exacerbates the problem. Several methods exist for mitigating numerical singularity including training data removal,[24] selecting sharper kernel functions,[11] and adding Gaussian noise to the kernel.[20,21] As we will show in section A-5, we are using Gaussian noise, not only for the stability it adds, but also for its ability to tolerate numerical inaccuracies that come with complex design problems.

3. Hyperparameter Selection

To use the kernel function, the hyperparameters θ_h must be chosen. Different values will yield different fits, each being a different view of the data. A common way to tune these parameters by maximizing Marginal Likelihood.[12] This is defined by

$$\log p(f_p|x_p,\theta_h) = -\frac{1}{2} f_p^\top [\sigma]^{-1} f_p - \frac{1}{2}\log|[\sigma]| - \frac{n}{2}\log\, 2\pi \tag{6}$$

Maximizing the marginal likelihood is itself an optimization problem and can be solved with a gradient based optimizer. This study used Matlab's sqp optimizer contained in fmincon and provided gradients using the complex step method. It was found that the solution was much more stable if constraints were placed on the hyperparameters based on information known about the design space. We describe this in section B-1.

4. Adding Gradient Information

Modeling the influence of gradients on the fit involves adding information to the covariance matrix. This requires finding a covariance function to model the correlation between points and derivatives. One approach to do this is shown for Co-Kriging by Chung[25] by deriving the covariance functions from the definitions of variance and derivative. Another approach suggested by Papoulis[26] and used for gradient enhanced GPR by Solak[9] exploits the theorem that the linear operation of an expected value is the expected value of the linear operation. The result from either approach is the same. To include gradient information in the fit, simply take the derivatives of the covariance function:

$$\begin{aligned} k\left(\tfrac{\partial p}{\partial x_v}, q\right) &= \left.\tfrac{\partial k(p,q)}{\partial x_v}\right|_q \\ k\left(p, \tfrac{\partial q}{\partial x_w}\right) &= \left.\tfrac{\partial k(p,q)}{\partial x_w}\right|_p \\ k\left(\tfrac{\partial p}{\partial x_v}, \tfrac{\partial q}{\partial x_w}\right) &= \left.\tfrac{\partial}{\partial x_w}\left(\left.\tfrac{\partial k(p,q)}{\partial x_v}\right|_q\right)\right|_p . \end{aligned} \tag{7}$$

The gradient information must be packed into the covariance matrix. This can be done by updating the definition for the covariance function as follows

$$k(p,q) \rightarrow \begin{bmatrix} k(p,q) & k\left(p, \frac{\partial q}{\partial x_w}\right) \\ k\left(\frac{\partial p}{\partial x_v}, q\right) & k\left(\frac{\partial p}{\partial x_v}, \frac{\partial q}{\partial x_w}\right) \end{bmatrix} \qquad f_p \rightarrow \begin{bmatrix} f_p \\ \frac{\partial f_p}{\partial x_d} \end{bmatrix}, \tag{8}$$

where submatrix $k(p,q)$ has dimension $n \times n$, and submatrix $k\left(\frac{\partial p}{\partial x_v}, \frac{\partial q}{\partial x_w}\right)$ has dimension $n\,d \times n\,d$. Important to note is that this matrix grows substantially with increasing design space dimension. This increases the cost of the response surface considerably. This cost can be offset by pre-computing the Cholesky decomposition of the covariance matrix, however, it is a constricting factor during hyperparameter learning process where the inversion must be completed several times to evaluate the marginal likelihood. As Forrester and others have identified, this is a major hurdle of stochastic surrogate-based optimization today.

5. Noise Models

It is possible to model several types of noise within the GPR framework.[12] The effect on the response surface formulation will have the form:

$$f_N^*(x) = f^*(x) + \epsilon, \tag{9}$$

where ϵ is a noise model. Adding noise to our model requires us to update our covariance matrix structure:

$$[k] \rightarrow [k] + [k_N], \tag{10}$$

where $[k]$ is the full covariance matrix for functions and gradients, and $[k_N]$ is the noise component of the covariance matrix.

A simple but useful model is an independent identically-distributed Gaussian noise with zero mean and given variance. This will only affect the self-correlated covariance terms along the diagonal of $[k_N]$. The noise covariance matrix will then take the form:

$$[k_N] = \begin{bmatrix} \theta_3^2 I_{n',n'} & 0_{n',m'} \\ 0_{m',n'} & \theta_4^2 I_{m',m'} \end{bmatrix}, \tag{11}$$

where I and 0 are identity and zero matrices with $n' = n(1+d)$ and $m' = m(1+d)$. Note we have allowed two separate noise hyperparameters for the function values and gradients. It is possible to individually train one noise parameter per gradient, however we have avoided this due to the numerical expense.

Adding this diagonal component to the covariance matrix relaxes the requirement that the fit exactly honors the training data. Depending on the magnitude of the noise hyperparameter, the fit will be allowed to stray a certain distance away from the data.

There are two main reasons for including noise in what would otherwise be considered a deterministic simulation. The first is to allow us to model noise in the objective function and gradients that may arise from numerical accuracies. Noise in the gradients is especially a concern for the motivating design problem as we are analyzing complex geometries with practical (coarse) tetrahedral cells on the body of an aircraft which generates strong shocks. Giles has identified this as a strong source of numerical error for the continuous adjoint formulation for such problems.[22] Second, even a small amount of noise, on the order of $\theta_1 \cdot 10^{-6}$, provides enough stability to the covariance matrix to prevent from becoming numerically singular. This allows the response surface to accept new training data that are close together, which is useful for targeting refinement near the estimated minimum.

B. Surrogate Based Optimization

We have developed a surrogate based optimization approach that follows the basic process flow shown in Figure 2. A summary of this process is described here, and detail on modifications to the traditional process are given in the following sections.

The process starts by choosing an initial set of design configurations with Latin Hypercube Sampling, which are evaluated using the SU^2 simulation suite. This involves deforming the baseline mesh to the desired configuration, performing the direct solution to solve for the objective and constraint function values, and then performing multiple adjoint solutions to solve for the objective and constraint gradients. The data is given to the response surface modeler, which must tune a set of hyperparameters for both the objective and constraint surfaces. This is the first of two internal optimization problems required by the SBO. Two sets of convergence monitors are then checked. The first is based on the expected improvement, which monitors the efficient global optimization of the response surface surface. The second is based on the estimated optimum, which attempts to refine the response surface's final estimate of the optimum. Depending on which criteria is converged, adaptive refinement is then performed by choosing a sampling criteria function and solving for its minimum using a combined genetic algorithm and gradient based optimizer. This is the second internal optimization problem. Once both convergence monitors reach a specified tolerance, the process ends as converged.

1. Robustness

Steps are taken to improve numerics and generality of the method by scaling the data based on the initial LHC sample. The sampled function value range, and design variable bounds are linearly scaled according to:

$$\begin{array}{llllll} f = f' \cdot f_{ref} + f_{off} & s.t. & [min(f), max(f)] & \rightarrow & [0,1] & \\ x_z = x'_z \cdot x_{z,ref} + x_{z,off} & s.t. & [min(x_z), max(x_z)] & \rightarrow & [0,1] & , z = 1, ..., d \\ \frac{\partial f}{\partial x_z} = \frac{\partial f'}{\partial x'_z} \cdot \frac{f_{ref}}{x_{z,ref}}. & & & & & \end{array} \tag{12}$$

Several benefits are realized from scaling the data past improving the condition of the covariance matrix, if we can assume the response surface is smooth with a nominal amount of variation. First, we can approximately say the variation of data is brought to be the same order of magnitude. This allows us to

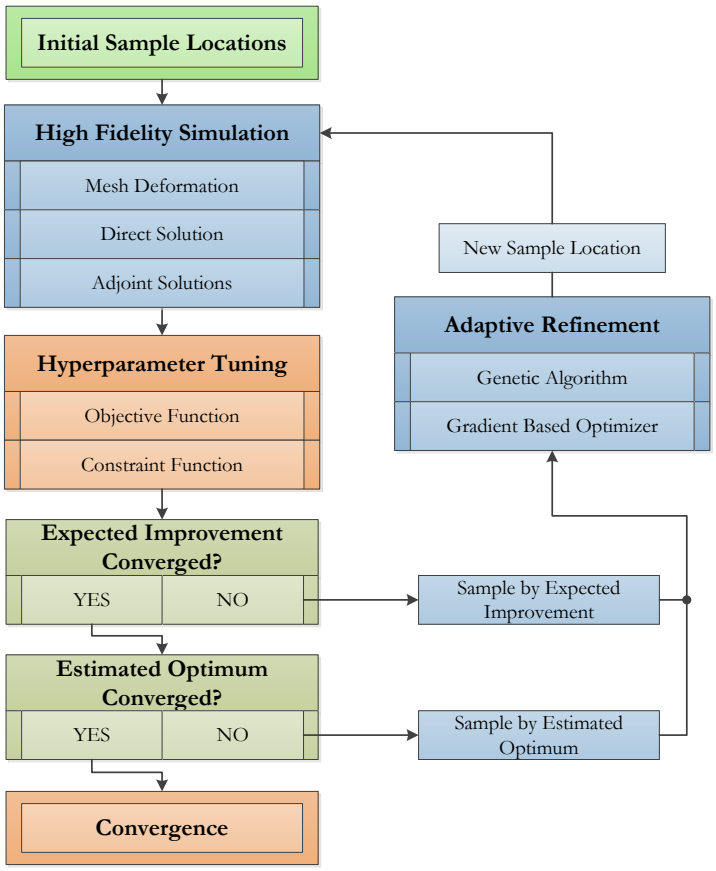


Figure 2: Surrogate Based Optimzation Process

assume isotropy of variation in each dimension and reduce the number of length scale parameters to one. This significantly reduces the computational cost of hyperparameter learning. Second, this allows us to claim that the scaled magnitude of the noise parameters for f and $\frac{\partial f}{\partial x}$ are of the same order of magnitude. This is important since the value of the noise in the gradients is difficult to estimate *a-priori*. Finally, it makes the problem robust to more design problems, where different design parameters of different scales can be mixed without having to learn separate length scales.

Further steps are taken to improve the robustness of the hyperparameter selection process by expressing additional *a-priori* knowledge in the form of nonlinear constraints. These constraints are summarized in the Table 1 below.

Table 1: Non-Linear Hyperparameter Constraints for MLE Minimization

Constraint	**Motivation**
$\frac{\theta_3}{\theta_1} < 1 \times 10^{-1}$	Avoid interpreting data as noise
$\frac{\theta_3}{\theta_1} > 1 \times 10^{-8}$	Maintain well conditioned numerics
$\text{rcond}(k(x_p, x_q)) > 1 \times 10^{-10}$	Maintain well conditioned numerics
$\theta_3 < \theta_4$	Trust objective function data before gradient data

In the above, $\text{rcond}(\cdot)$ is the reciprocal condition number. These constraints add robustness to the learning process by encouraging not only favorable noise hyperparameters, θ_3 and θ_4, but also indirectly control the nominal variance and length scale parameters θ_1 and θ_2. For example, using a small noise parameter can

encourage the marginal likelihood optimization to choose a small length scale, which in turn results in a response surface that is mostly flat with sharp peaks at the training data. The fourth constraint $\theta_3 < \theta_4$ is a safe guard against the gradient data from overpowering the behavior of the fit enough that it does not respect the function data. This can happen when using gradients from continuous adjoints because they are not the numerically exact derivative to the direct solution (the discrete adjoint is however)

Additionally, bound constraints are enforced on the hyperparameters to ensure that they stay within values relevant to the sampled data. These constraints are shown in table 2

Table 2: Hyperparameter Bound Constraints for MLE Minimization

Bounds	**Description**
$\min(\Delta f_i)/W < \theta_1 < \max(\Delta f_i) \cdot W$	Bound by minimum and maximum distance between function values
$\min(\Delta x_i)/W < \theta_2 < \max(\Delta x_i) \cdot W$	Bound by minimum and maximum distance between data locations

These bounds are chosen to avoid attempting to model more, or less, variation than the training data could hope to predict. This is why they are tied to the maximum and minimum observed variation in function value and data location. In the above, W is constant chosen to allow larger or smaller values. A value of 100 was used for W because these bounds were used only to constrain θ_1 and θ_2 to reasonable orders of magnitude.

2. Adaptive Refinement

Our approach to surrogate based optimization involves three sampling stages - first an initial sample, second an efficient global refinement, and last a refinement of the estimated minimum. These stages are designed to efficiently dissect the design space and attempts to yield the best estimate of the optimum only in the last sample. The main motivation of this construction is to build in a sense of convergence that is similar to that of gradient-based optimization.

First, the design space is sampled with a set of well-spread initial design points chosen with Latin Hypercube (LHC) sampling restricted to the hypercube,

$$x_k^l < x_k < x_k^u \ , \ k = 1, ..., d. \tag{13}$$

The initial $x = 0$ design is included with this sample as the first point. Given enough computational resources, this first phase is nicely parallelizable as each design point can be run independently. The number of samples to take should depend on the expected complexity of the design space. For the problems explored in this study, two samples per dimension has worked well.

Second, the surrogate model is refined using an Infill Sampling Criteria (ISC) based upon Expected Improvement (EI). This sampling criteria is well known for its ability to leverage both estimated function value and uncertainty to balance design space exploration (targeting global accuracy) and exploitation (targeting local optimality).[20,27] Expected improvement is commonly expressed as:

$$\begin{aligned} \mathrm{E}\left[I(x)\right] &= \mathrm{E}\left[\max(f_{\min} - F, 0)\right] \\ &= (f_{\min} - f^*)\Phi\left(\tfrac{f_{\min} - f^*}{s^*}\right) + s^*\phi\left(\tfrac{f_{\min} - f^*}{s^*}\right). \end{aligned} \tag{14}$$

In the above, $\Phi(\cdot)$ and $\phi(\cdot)$ are the standard normal distribution and density functions, $s^* = \mathbb{V}[f^*(x)]$ is the estimated variance of the response surface, and $f_{\min}$ is the minimum of the current training point sample.

If additional information in the form of constraints is available in the problem, it can be added to the infill sampling criterion by searching in locations of high probability of feasibility. Following the method described by Shimoyama,[23] this is done by building a second surrogate model and evaluating the expected improvement, conditioned by the probability of feasibility,

$$\mathrm{P}[c(x) < 0] = \phi\left(\frac{c^*}{s_c^*}\right), \tag{15}$$

where the constraint is formulated as $c(x) < 0$, and c^* is the estimated constraint value with estimated variance, s_c^*.

In our experience, the expected improvement infill sampling criteria often added training data near the box boundaries and corners of the design space. This is inefficient because at least half of a training point's region of influence is wasted modeling the response surface outside the boundaries. To encourage sampling inside the design space, we apply a penalty function,

$$\mathrm{B}(x) = 1 - \exp\left(-\frac{1}{2}\min\left(\frac{x_k - x_k^l}{b_k^2}, \frac{x_k^u - x_k}{b_k^2}, k = 1, ..., d\right)\right), \tag{16}$$

which is a Gaussian function of the distance from the nearest hypercube boundary. In this study, the length-scale b_k is set to $0.1(x_k^u - x_k^l)$, or 1% of hypercube dimension k's length. The above penalty function is greatly simplified in the non-dimensional space constructed in the previous section.

Combining the Expected Improvement, Constraint Penalty, and Boundary Penalty results in the first infill sampling criterion:

$$\begin{aligned} ISC_1(x) &= \mathrm{E}\left[I(x)\right] \cdot \mathrm{P}[c(x) < 0] \cdot \mathrm{B}(x) \\ x_{new} &= x \,|\, \max\left(ISC_1(x)\right), \end{aligned} \tag{17}$$

The response surface is refined by sampling $f(x_{new})$ and $c(x_{new})$ in this way until the criterion converges to a small value,

$$R_1(i) = ISC_1(x_i) < T_1, \tag{18}$$

where T_1 is a specified tolerance.

While indeed efficient, our experience with GPR-based SBO has suggested that after expected improvement has converged to some preset value, it becomes difficult to discover the sharp and narrow wells that develop in its surface. For this reason, it can be more efficient to switch to an infill sampling criteria based on pure exploitation.

Thus the final phase of our method's design space sampling is based on refining around the estimated deterministic optimum of the surrogate model:

$$\begin{aligned} ISC_2(x) &= f^*(x), \\ x_{new} &= x \,|\, \min\left(ISC_2(x)\right), \;\; s.t. \;\; c^*(x) < 0. \end{aligned} \tag{19}$$

This procedure continues until the measured improvement of the design sample converges to a small value

$$R_2(i) = \max\left(\, f_{best} - f_i \,,\, R_2(i-1) \,\right) < T_2, \tag{20}$$

or until the sampling stalls after not finding an improvement for a predetermined number of iterations. In the above equation, T_2 is another specified tolerance, and f_{best} is the minimum of the training data that came before f_i. We use similar monitors on the change in location of the best point, and the norm of the gradient of the best point, but often the function value of best point converges the fastest.

Both the second and third sampling stages use a two-part optimization of the surrogate model. First a genetic algorithm is used to find a reasonable estimate of the global minimum. This estimate is fed as the initial guess to a gradient based optimizer, which returns a better estimate of the global minimum. This process requires several thousand function evaluations of the response surface, but is far less expensive than evaluating an additional high-fidelity design point.

C. Methodology: Gradient Computation Using Continuous Adjoints

The objective of this section is to describe the way in which we quantify the influence of geometry modifications on either the pressure distribution on the aircraft surface or the equivalent area distribution.

The fluid domain Ω is bounded by a disconnected boundary $\delta\Omega$ which is divided into a "far-field" component, Γ_∞, and a solid wall boundary, S. Ω has been further divided into two subdomains Ω_i and Ω_o separated by the "near-field" boundary Γ_{nf}. Note that Γ_{nf} will remain fixed throughout the optimization process, but the solid surface S will change as needed to meet the optimization criteria.

A typical optimization problem seeks the minimization of a certain cost function J with respect to changes in the shape of the boundary S. We will concentrate on functionals defined as integrals over the solid surface

S, and integrals over the near-field boundary Γ_{nf},

$$J = \int_S \vec{d}\cdot(P\vec{n}_S)\,ds + \int_{\Gamma_{nf}} g(x,P)\,ds = \int_S j_S\,ds + \int_{\Gamma_{nf}} j_{nf}\,ds, \tag{21}$$

where P is the value of the pressure, $\vec{n}_S$ is a normal vector to the solid surface S pointing outside the domain, $g(x,P)$ is a function that depends on the spatial coordinates and the pressure at the near-field, and $\vec{d}$ is an arbitrary constant vector that we will define later on. The goal is to compute the variation of the above functional caused by arbitrary (and multiple) deformations of S.

Upon an infinitesimal deformation δS of the control surface S along the normal direction $\vec{n}_S$, the cost function varies due to the changes in the solution induced by the deformation:

$$\delta J = \int_S \vec{d}\cdot\delta P\vec{n}_S\,ds + \int_{\Gamma_{nf}} \frac{\partial g(x,P)}{\partial P}\delta P\,ds + \int_S (\vec{d}\cdot\vec{\nabla}P)\delta S\,ds. \tag{22}$$

In the last expression we note that the objective function variation depends on the linearized steady Euler flow equations, $\vec{\nabla}\cdot(\vec{\delta}F) = 0$, where $\vec{F}$ are the convective fluxes. And we resort to the adjoint state $\Psi = (\psi_1, \vec{\varphi}, \psi_5)$ to tackle the variation of the flow variables δP in the objective function:

$$\begin{aligned}\delta J &= \int_S (\vec{d}\cdot\vec{n}_S)\delta P\,ds - \int_S (\vec{n}_S\cdot\vec{\varphi})\delta P\,ds + \int_{\Gamma_{nf}} \frac{\partial g(x,P)}{\partial P}\delta P\,ds - \int_{\Gamma_{nf}} \Delta\Psi\left(\vec{n}_{nf}\vec{A}\right)\delta U\,ds \\ &+ \int_S \left(\vec{d}\cdot\vec{\nabla}P + (\partial_n\vec{v}\cdot\vec{n}_S)\,\vartheta + \nabla_S(\vec{v}\,\vartheta)\right)\delta S\,ds,\end{aligned} \tag{23}$$

where $\Delta\Psi = \Psi_i - \Psi_o$ is the difference between the values of Ψ above and below the near-field boundary, $\vec{A}$ is the jacobian matrix of the fluxes, and $\vartheta = (\rho\psi_1 + \rho\vec{v}\cdot\vec{\varphi} + \rho H\psi_5)$. Finally, the following adjoint system should be solved

$$\vec{A}^T\cdot\vec{\nabla}\Psi = 0. \tag{24}$$

where the boundary equations will be discussed above.

Using a pressure base functional on the surface, the vector $\vec{d}$ is defined as

$$\vec{d} = \begin{cases} \left(\frac{1}{C_\infty}\right)(\cos\alpha\,\cos\beta, \sin\alpha\,\cos\beta, \sin\beta), & C_D \quad \text{Drag coefficient,} \\ \left(\frac{1}{C_\infty}\right)(-\sin\alpha, \cos\alpha, 0), & C_L \quad \text{Lift coefficient,} \end{cases} \tag{25}$$

where α is the angle of attack, β is the angle of sideslip, $C_\infty = \frac{1}{2}v_\infty^2\rho_\infty A_z$, and v_∞ and ρ_∞ denote the infinity values of velocity and density. Using this particular definition of $\vec{d}$, the boundary condition for the adjoint equations are:

$$\vec{n}_S\cdot\vec{\varphi} = \vec{d}\cdot\vec{n}_S, \quad \text{on } \Gamma_S \tag{26}$$

Apart from surface pressure based functionals, in this particular work, the equivalent area distribution is used in the optimization problem. The equivalent area distribution $A_e(x,\theta)$ at a particular azimuthal angle θ is given by

$$A_e(x,\theta) = \int_0^x C(P - P_\infty)(x-t)^{1/2}\,dt, \tag{27}$$

where $C = 4\frac{\sqrt{2\beta r}}{\gamma P_\infty M_\infty^2}$, and the line integral is evaluated on a near-field line characterized by an azimuthal angle θ. In order to pose an inverse design problem whereby we attempt to match a target equivalent area distribution, the logical approach is to formulate the problem using a least-squares minimization formulation. Let's minimize the weighted sum of the square of the differences between the computed equivalent area and the target equivalent area at N different points x_i $(i = 0, ..., N-1)$, and different azimuthal angles $k = 0, ..., M$, with weights ω_{ik}, and target equivalent areas $A_t(x_i, \theta_k)$. The objective function can be written as:

$$J = \sum_{k=0}^{M}\sum_{i=0}^{N-1} \omega_{ik}\left[A_e(x_i,\theta_k) - A_t(x_i,\theta_k)\right]^2, \tag{28}$$

The variation of Eq. 28 is given by:

$$\delta J = \sum_{k=0}^{M} \sum_{i=0}^{N-1} 2\omega_{ik} \left[A_e(x_i, \theta_k) - A_t(x_i, \theta_k) \right] \delta A_e(x_i, \theta_k). \tag{29}$$

On the other hand, the variation of the equivalent area, $A_e(x, \theta)$, with respect to the pressure yields:

$$\delta A_e(x, \theta) = \int_0^x C(x - t)^{1/2} \delta P \, dt. \tag{30}$$

Using this last expression it is possible to separately evaluate the variation of the non-zero equivalent areas at the chosen points as follows:

$$\delta A_e(x_i, \theta) = \sum_{j=0}^{i-1} \int_{x_j}^{x_{j+1}} C(x_i - x)^{1/2} \delta P \, dx, \tag{31}$$

where, for the sake of simplicity, we have substituted the dummy variable of integration t by x. Using the variation of the equivalent area that we have computed in Eq. 29, it is possible to rewrite the variation of the objective function as:

$$\delta J = \sum_{k=0}^{M} \sum_{i=0}^{N-1} \left[2\omega_{ik} \left[A_e(x_i, \theta_k) - A_t(x_i, \theta_k) \right] \sum_{j=0}^{i-1} \int_{x_j}^{x_{j+1}} C(x_i - x)^{1/2} \delta P \, dx \right], \tag{32}$$

where, to simplify the notation, we will use:

$$\Delta A_e(x_i, \theta_k) = 2C\omega_{ik} \left[A_e(x_i, \theta_k) - A_t(x_i, \theta_k) \right]. \tag{33}$$

Rearranging the integrals in Eq. 32, it is possible to write the variation of the objective function as

$$\delta J = \sum_{k=0}^{M} \sum_{j=0}^{N-2} \int_{x_j}^{x_{j+1}} \sum_{i=j+1}^{N-1} \left(\Delta A_e(x_i, \theta_k)(x_i - x)^{1/2} \right) \delta P \, dx. \tag{34}$$

Using this final expression, the admissible adjoint boundary condition that eliminate the dependence on the fluid flow variation in the inverse equivalent area shape design problem is:

$$\vec{\nabla}\Psi \left(\vec{n}_{nf} \cdot \vec{A} \right) = h(x, \theta, P), \quad \text{on } \Gamma_{nf} \tag{35}$$

and the function $h(x, \theta, P)$ is defined at each discrete azimuthal angle θ_k as

$$h(x, \theta_k, P) = \begin{cases} 0 & , \text{ if } \quad -L < x < x_0, \\ \sum_{i=j+1}^{N-1} \Delta A_e(x_i, \theta_k)(x_i - x)^{1/2} & , \text{ if } \quad x_0 \leq x \leq x_{N-1}, \\ 0 & , \text{ if } \quad x_{N-1} < x < L, \end{cases}$$

where x_j is the closest discrete lower value to x.

III. Results

A. Analytical Test Cases

The following examples attempt to minimize the Rosenbrock function,

$$f(x) = \sum_{i=1}^{d-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right], \tag{36}$$

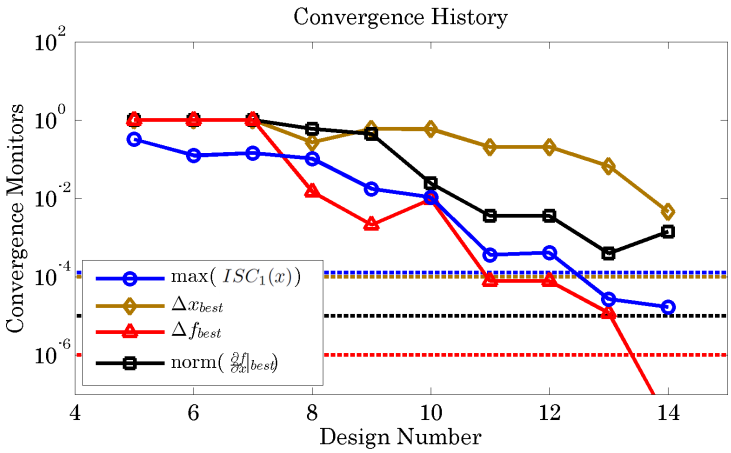
with and without the Rastrigin function as a constraint,

$$c(x) = 10n + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i)\right] - c_o < 0 \tag{37}$$

where c_o was chosen to be 20 for this problem. These two functions (and their analytical gradients) exercise SBO's ability to handle complex and multimodal functions with constraints.

1. *2DV Rosenbrock Function: Unconstrained Minimization*

A plot of the convergence monitors for the surrogate based optimization of a two-dimensional Rosenbrock function is shown in Figure 3 below.



$$x_{min} = [1.001556, 1.002614]\ ,\ f_{min} = 0.000027$$

Figure 3: Convergence of Unconstrained 2D Rosenbrock Problem

This figure shows four convergence parameters. The first is the maximum modified expected improvement (R_1 as described earlier). The second is the norm of the change in location of the best training point, the third the magnitude of change in value of the best training point, and the fourth is the norm of the gradient of the best training point. The tolerances for convergence are plotted as the horizontal dashed line with corresponding color. For this problem, all four parameters converge quickly. The modified expected improvement residual converges in 12 iterations, and two additional iterations are needed to converge the objective function value.

The surrogate-based optimization was compared to a gradient based optimization with a start point at $x = [-1\,,\,1]$. A modified newton method (matlab's fmincon, trust-region-reflective method) was used and a comparison of the solution history is shown in Figure 4 .Three important comparisons can be drawn from this study. First, for this problem it took 68 fewer function evaluations of the test function to converge. The gradient based solution took so long because the initial point was placed such that the optimizer had to travel through the Rosenbrock function's long and flat banana-shaped valley, admittedly a contrived but typical case. However, this highlights the reduction in expensive function evaluations that can come by performing a thorough optimization on an accurate surrogate model. Second, the SBO was only able converge the error of

the function minimum to 10^{-5}, while the GBO was able to converge to 10^{-10}. This is because the surrogate model's accuracy is approximately limited by the value of the noise hyperparameters, which are still needed for this problem to maintain numerical stability. Finally, it is important to note that the GBO is fast in physical time, taking a fraction of a second to complete. Because the SBO is performing two secondary optimization problems at each external iteration however, it took approximately 400 seconds to converge this problem on a Intel quad-core PC with 8GB of ram. This is the trade-off made to build and scour the response surface model.

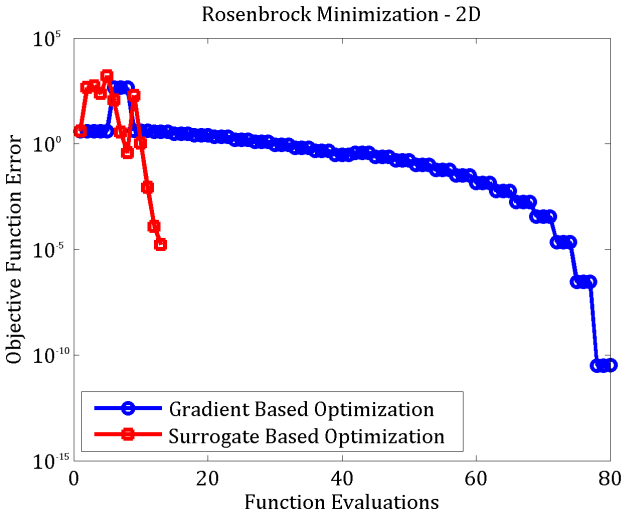


Figure 4: Comparison of Unconstrained GBO and SBO

2. 2DV Rosenbrock Function: Rastrigin Function Constrained Minimization

The surfaces plotted in Figure 5 show the information used by the surrogate based optimizer midway through the solution of the constrained problem. Despite being only partially constructed, the constraint surface in Figure 5b is able to provide enough information to identify probable regions of feasibility in Figure 5d. Along with the boundary penalty constructed to avoid sampling data in corners of the design space, this is used to condition the expected improvement, and the resulting surface is shown in Figure 5c. Solving for the maximum of the conditioned expected improvement identifies a suitable location to run an additional sample.

This solution is again compared to the gradient-based solution, which this time starts in a feasible region that does not contain the global optimum. Because the SBO is exploring the entire design space, it is able to find the global optimum.

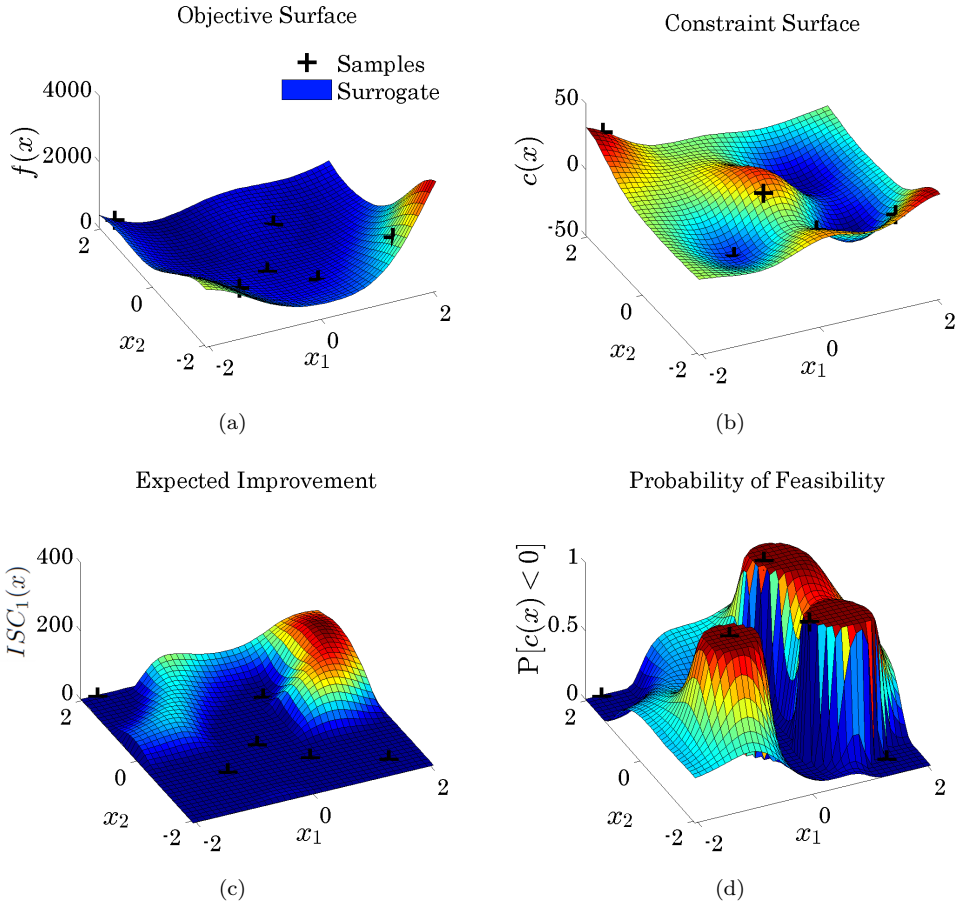


Figure 5: Example Learning Surfaces made with 6 Samples

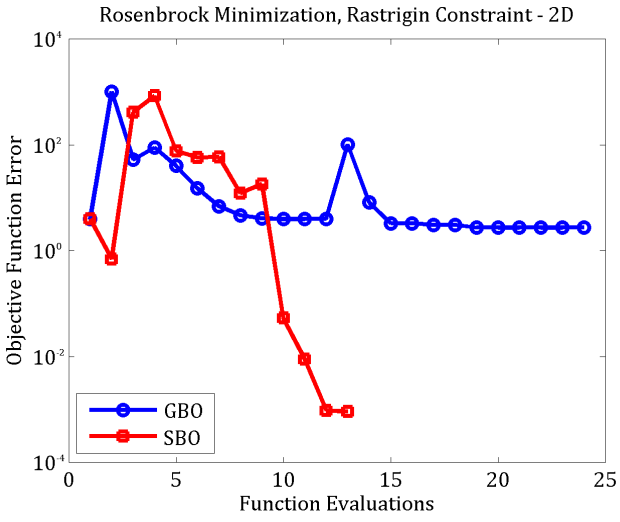


Figure 6: Comparison of Constrained GBO and SBO

B. 2D Euler Test Cases

1. *2DV Subsonic NACA0012 Airfoil: Lift Constrained Drag Minimization*

In this test case, the drag, lift and their gradients were estimated on a two-dimensional NACA0012 airfoil at an angle of attack of 1.25° and a Mach number of 0.8. Two Hicks-Henne bump functions were used for design variables, one each on the upper and lower surfaces. In this design, the optimum is located on the constraint boundary. The sampling criteria was able to populate several designs near the constrained optimum. Because this was a two dimensional design problem however, the SBO was only able to find two counts of drag reduction.

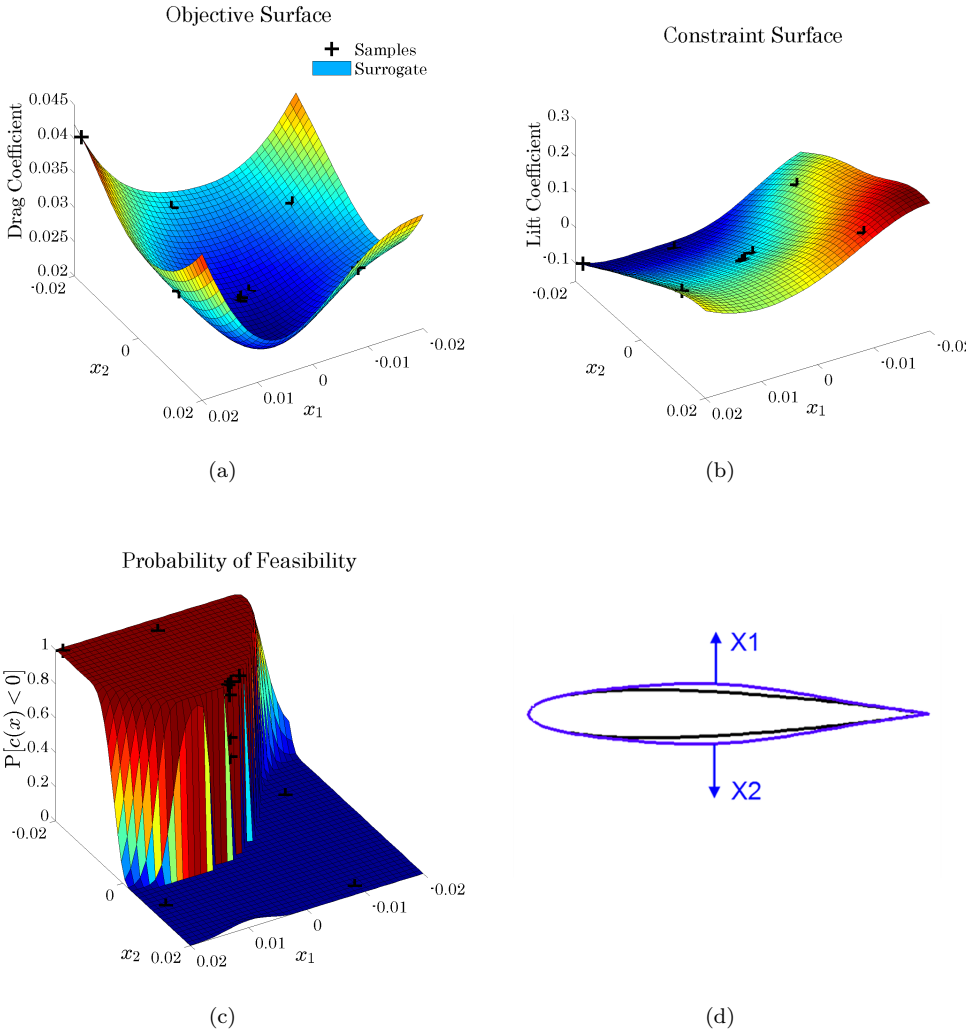
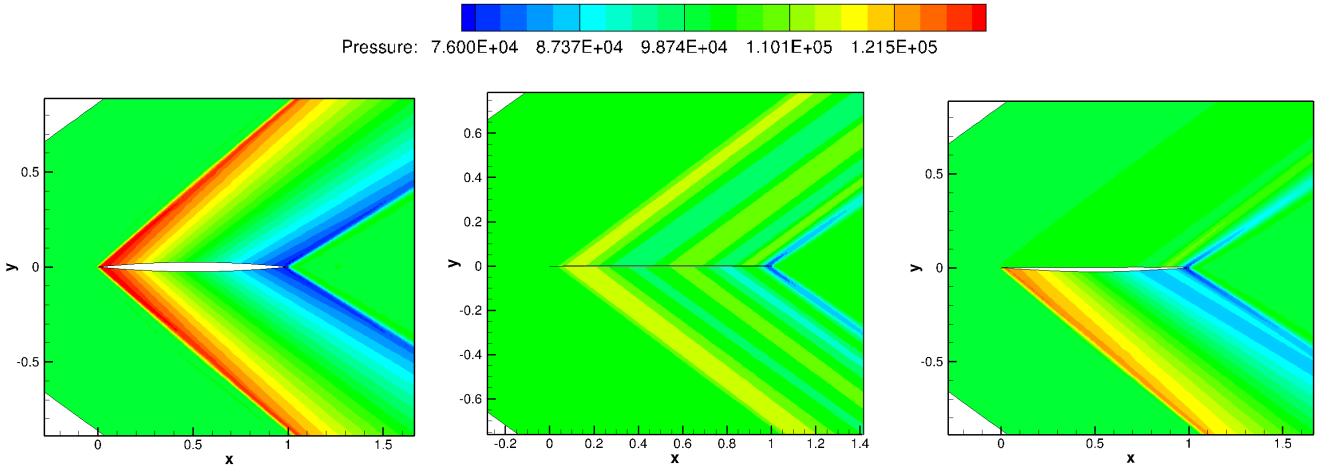


Figure 7: Learning Surfaces, NACA0012 Lift Constrained Drag Minimization

2. 10DV Supersonic Parabolic Airfoil: Equivalent Area Constrained Drag Minimization

In this test case, drag was reduced on a 5% thick parabolic airfoil at Mach 1.7 and no angle of attack, with and without a constraint on the equivalent area two body lengths below. The result for the unconstrained problem was to reduce the airfoil to a thin plate, which changes the equivalent area significantly. In the constrained problem, the equivalent area is sensitive only to the lower surface, which must remain unchanged. This leaves room however for drag reduction by flattening the upper surface.



(a) Baseline (b) Unconstrained SBO (c) A_e Constrained SBO

Figure 8: Biparabolic Airfoil Pressure Contours

As Figures 8a - 8c show, the constrained problem was able to partially enforce the constraint an maintain a similar pressure distribution underneath the the airfoil, while flattening the upper airfoil surface. Figure 9 compares the pressure signatures at the nearfield for the three cases. For the unconstrained case, a total of 25 design iterations were performed. For the constrained case, a total of 34 design iterations were performed.

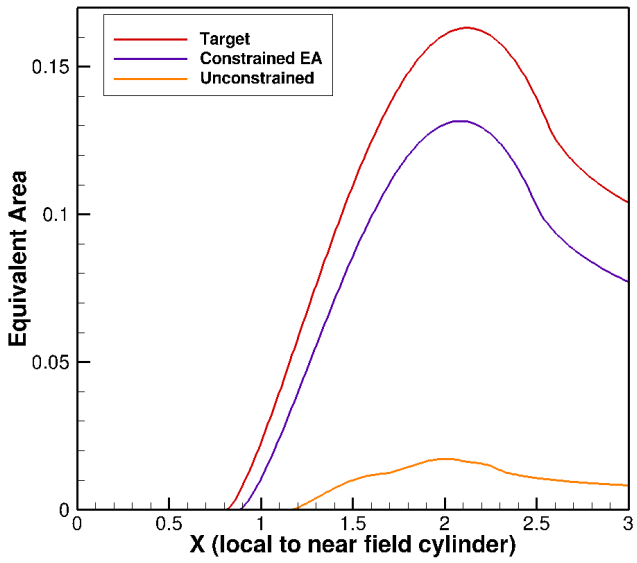


Figure 9: Equivalent Area Distributions for Constrained and Unconstrained Drag SBO

The convergence plot for this problem shows that the two infill sampling criterion are performing as designed. First, the plot of the convergence monitors below show that the value of the expected improvement converged quickly by iteration 24. If this criterion alone was used to refine the response surface, the optimization would have stalled very early with a large constraint violation. However, the second sampling criterion was able to start refining the estimate of the constrained optimum, and was able to reduce the constraint violation to a pre-set threshold of five percent.

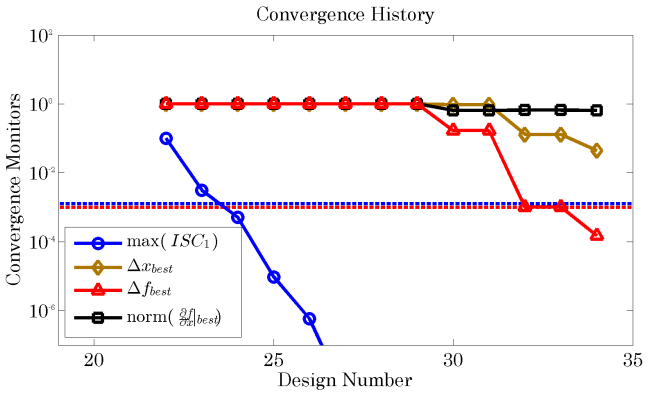
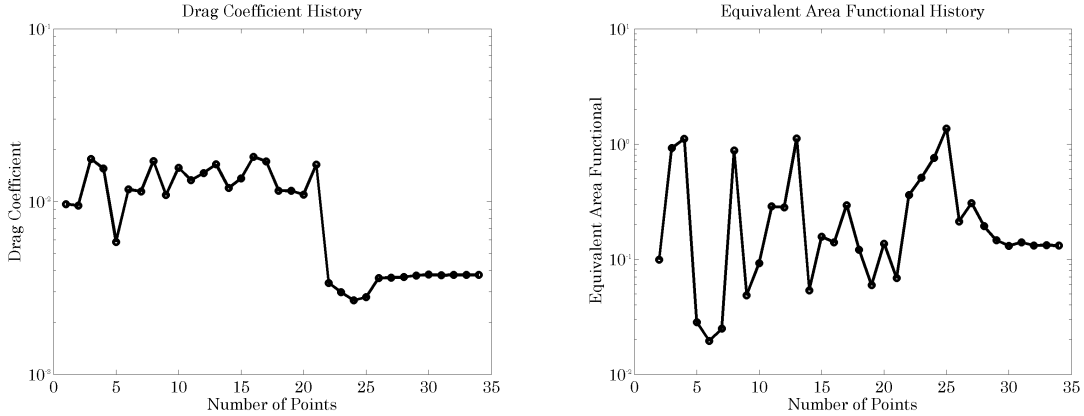


Figure 10: Convergence History: Biparabolic Airfoil



(a) (b)

Figure 11: Objective and Constraint History: Biparabolic Airfoil

The surrogate-based optimization approach struggles with this case, presumably because it violates our assumption of isotropic variation in the scaled design space. Because the equivalent area is not sensitive to the design variables above the airfoil, the true constraint surface will be exactly flat in the corresponding dimensions. The surrogate model, however, produces small variations in the constraint along these dimensions which requires the optimizer to explore many more designs to identify the true constraint boundary. This could of course be accounted for by tuning one length scale hyperparameter per dimension, at increased computational cost. A more clever approach might be to acknowledge *a-priori* the lack of sensitivity in these design variables and pre-scale their length scales by a large number. This would encourage the surrogate to more accurately model the low variation in these dimensions.

C. 3D Euler Test Cases

1. *LMCO N+2 Aircraft: Mesh and Geometry Setup*

Two studies were performed on the Lockheed N+2 aircraft baseline geometry shown in Figure 12.

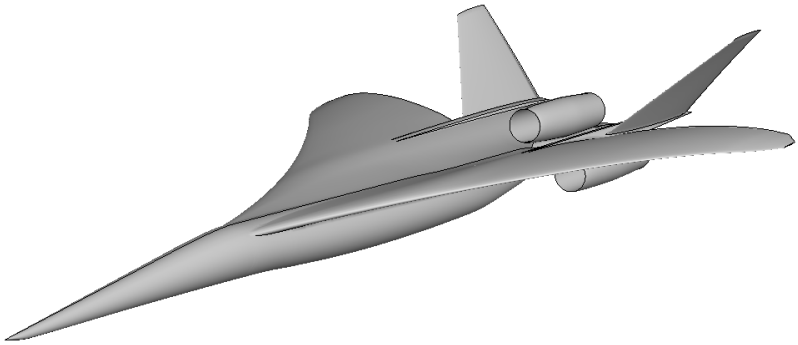
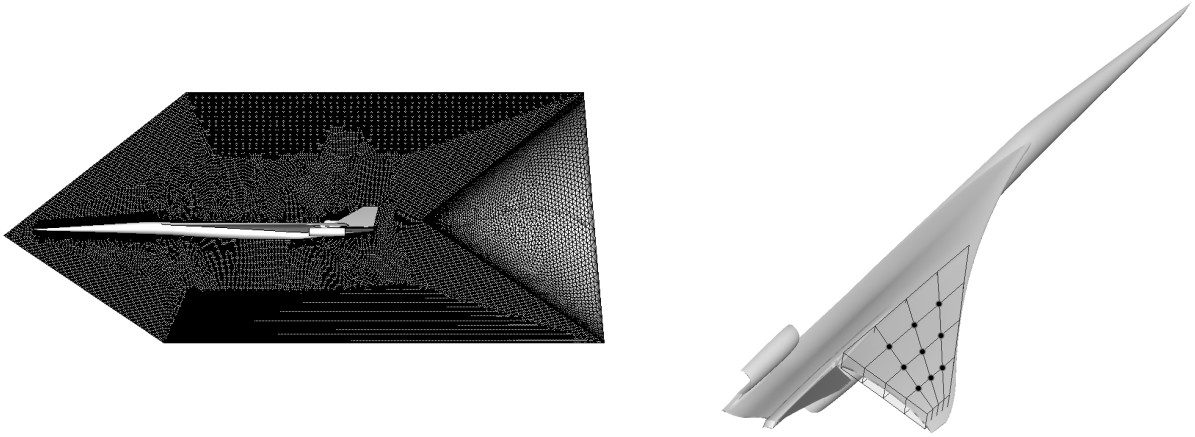


Figure 12: Baseline LMCO N+2 Supersonic Passenger Jet Geometry

The first was an unconstrained drag minimization using SBO. For this we used a 1.3 million node mesh without a near-field region, shown in Figure 13a, with nine free-form deformation design variables spread across the top surface of the wing, shown Figure 13.



(a) Drag Mesh

(b) Control Point Locations

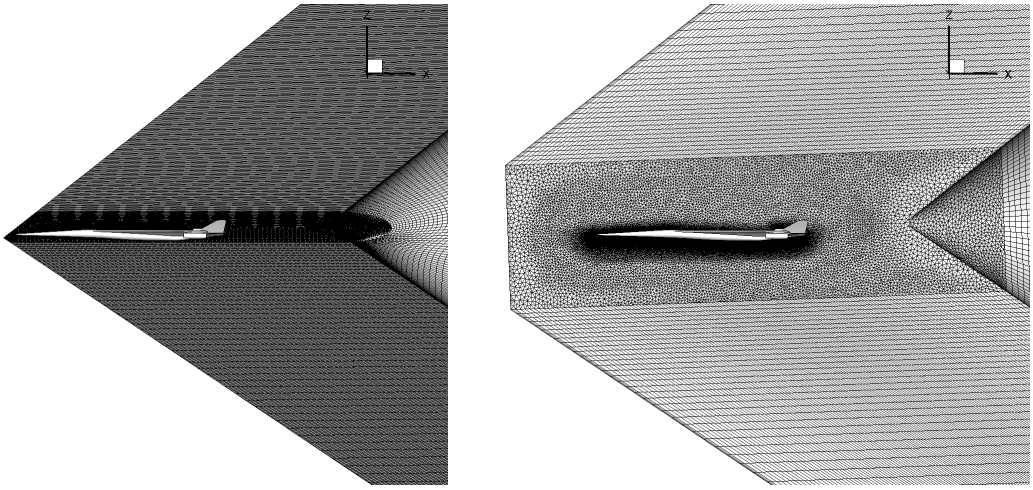
Figure 13: Mesh and Design Variables: N+2 Drag Minimization

The second was a multi-objective minimization of drag and target equivalent area violation using GBO. The multi-objective problem was posed as a combined objective problem with,

$$J = \alpha\, C_d + (1 - \alpha)\, \Delta A_e, \tag{38}$$

where ΔA_e is the target equivalent area objective function given in Equation 28. For this problem, we found that an α of $1 - 10^{-5}$ was suitable.

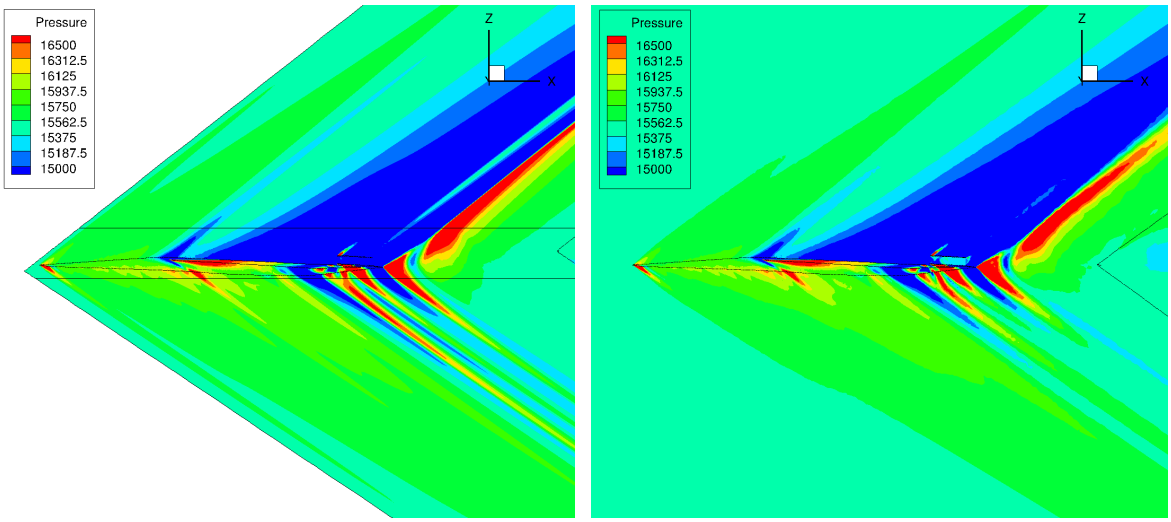
This second optimization problem requires a larger mesh that is able to resolve the nearfield, (which we located two body-lengths below the aircraft). To reduce the computational expense, we coarsened the resulting mesh from 3.1 million nodes to 1.2 million nodes (4.3 million cells). Nodes were also added in front of the aircraft to better estimate the adjoint. A comparison of the coarse and fine grid is shown in Figure 14.



(a) Fine Grid, 3.1 million nodes (b) Coarse Grid, 1.2 million nodes

Figure 14: Fine and Coarse Boom Optimization Meshes

A refinement study verified that while the coarse mesh is smoothing out the pressure signature, it is still able to capture the same basic characteristics, such as the initial pressure rise, and strong shocks from the main wing's trailing edge. A comparison of the pressure field on the symmetry plane in Figure 15 shows similar behavior between the two meshes. Figure 16 compares the pressure signature shows the similar behavior in near field pressure signature.



(a) Fine Grid, 3.1 million nodes (b) Coarse Grid, 1.2 million nodes

Figure 15: Pressure Contours for Fine and Coarse Meshes

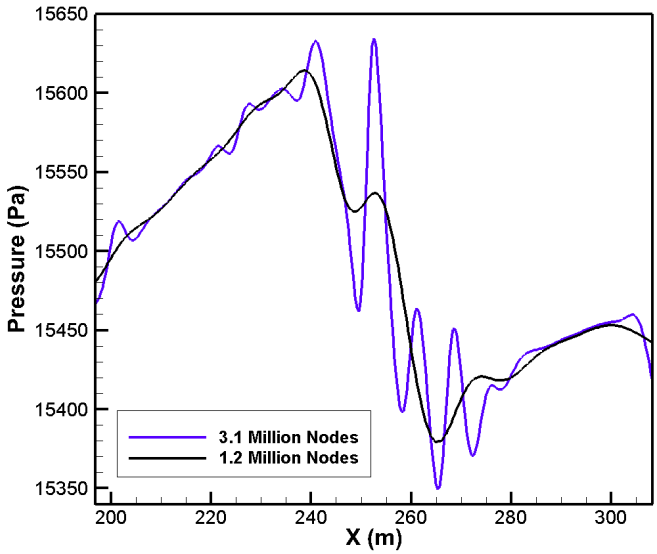


Figure 16: Comparison or Near Field Pressure Signatures for Coarse and Fine Mesh

Eleven design variables for this problem were spread across the upper and lower wing surfaces, as well as underneath the fuselage, as shown in Figure 17. All eleven were defined to deform the mesh in the vertical direction. Together these variables allow changes in camber and thickness of the wing and thickness of the fuselage, in order to allow the optimizer to trade lift and thickness to maintain the target equivalent area distribution.

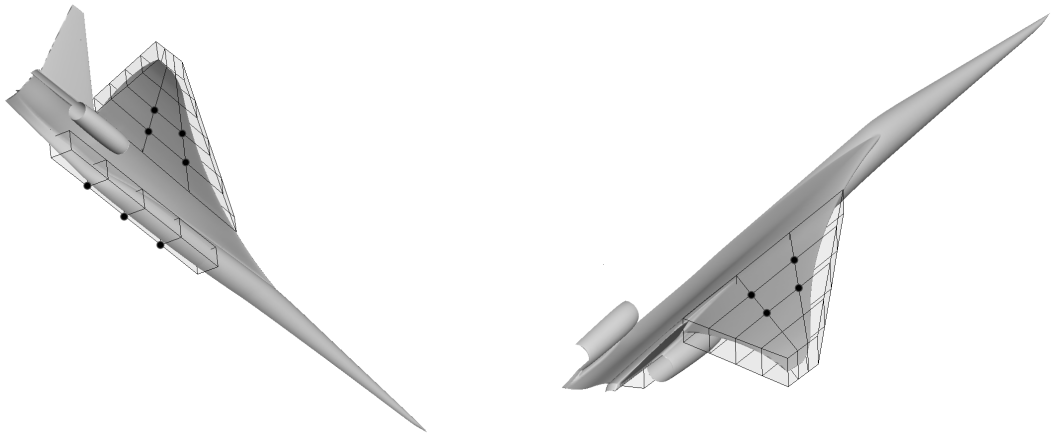


Figure 17: FFD Control Points: N+2 Boom Minimization

2. *9DV N+2 Aircraft: Unconstrained Drag SBO*

A surrogate based optimization was performed on the N+2 using the nine free-form deformation control points placed on upper wing surface. A comparison with a gradient-based optimization of the problem is shown in Figure 18. The surrogate based optimizer was able to discover a 4.6% reduction in drag in 20 fewer iterations than a gradient based optimizer. Also, because the first 21 design points are independent of each other, they can be simulated simultaneously given enough computing resources, which further reduces the physical time spent optimizing.

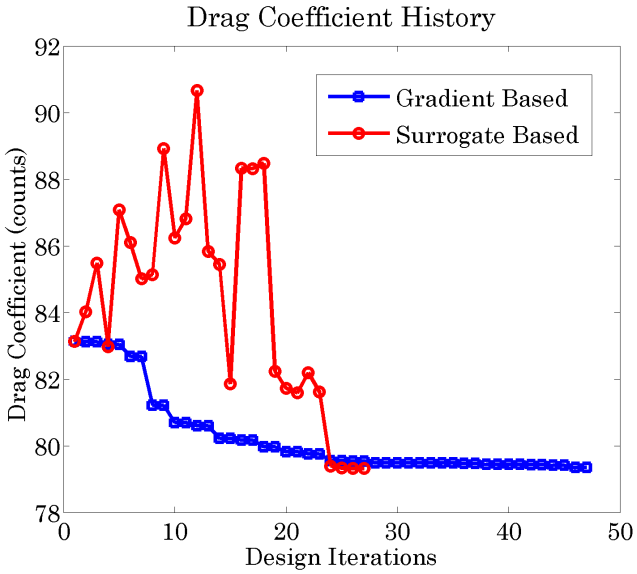


Figure 18: Comparison of Unconstrained Drag GBO and SBO

A comparison of the geometry change between the baseline and final design in Figure 19 shows that the drag reduction was accomplished by reducing the thickness of the wing.

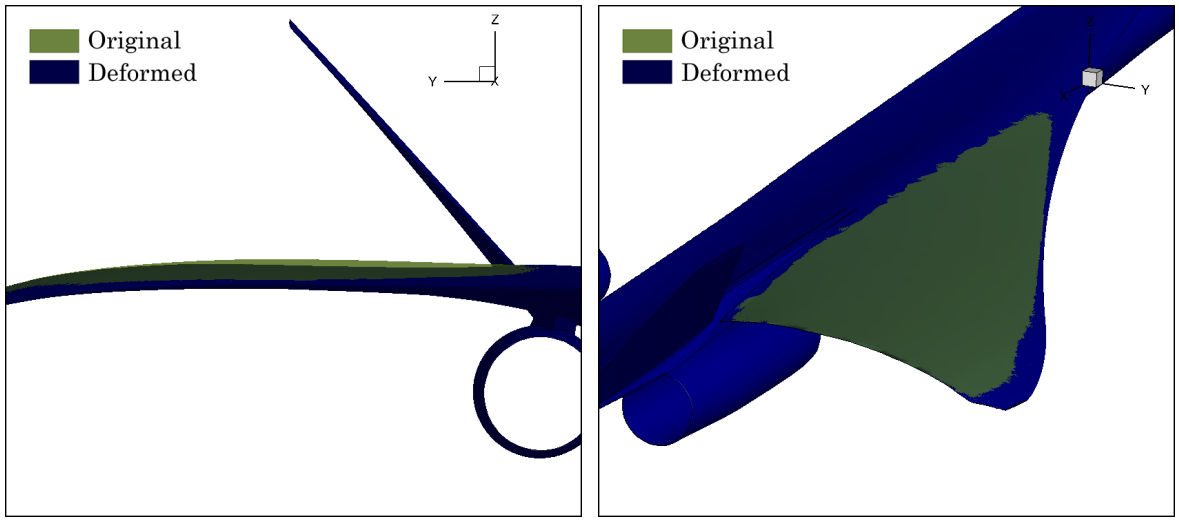
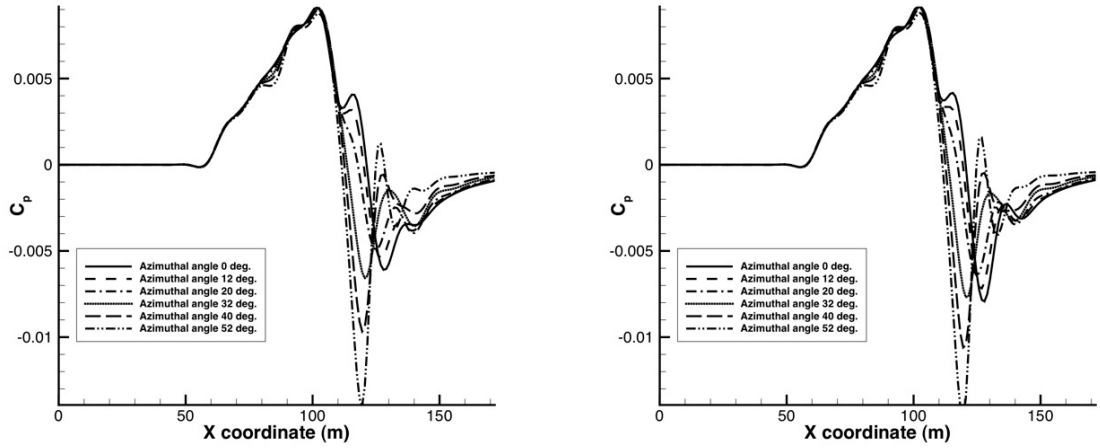


Figure 19: Original and Deformed N+2 Surfaces, Unconstrained Drag Minimization

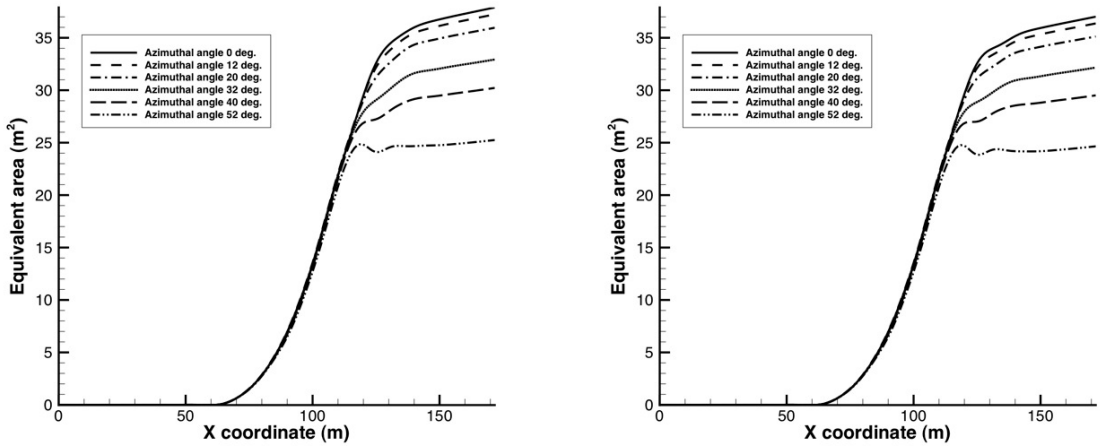
3. 11DV N+2 Aircraft: Multi-Objective Equivalent Area and Drag GBO

Work is on-going to combine the new response surface methodology with the gradients available from the new adjoint formulation for target equivalent area functional. To demonstrate the ability of the adjoint formulation to enable design optimization while maintaining a target equivalent area distribution across multiple azimuths, we performed a gradient-based multi-objective optimization of the N+2. The optimization was performed for nine iterations with azimuths varying from 0° to 60° (pressure disturbances above 60° do not reach the ground). The drag was reduced from 0.00875 to 0.00850, a 3% reduction. A comparison of the near-field pressure distributions and equivalent areas between the baseline and final design are shown in Figures 20 and 21. The plots for equivalent area distribution show at most a 2% change from the target.



(a) Baseline (b) Final

Figure 20: Baseline and Final Near-field Pressure Signatures



(a) Baseline (b) Final

Figure 21: Baseline and Final Target Equivalent Area Distributions

A comparison of the baseline and final geometry in Figure 22 showed that the upper wing surface was flattened, and lower fuselage deformed inwards in order to reduce the drag. To compensate for the fuselage's

effect on the equivalent area, the lower wing surface was deformed downwards as well. A comparison of surface pressures for the design study are given in Figure 23, and contours of the density and adjoint density fields are shown in Figure 24 for visualization.

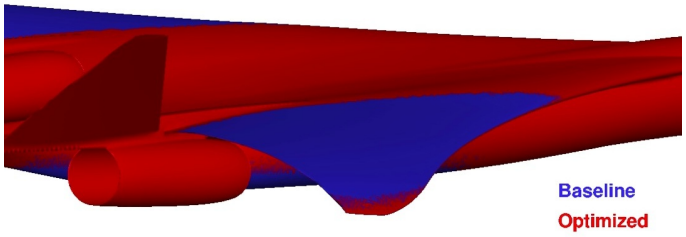


Figure 22: Baseline and Final N+2 Geometry Comparison (Mirrored Full-Body)

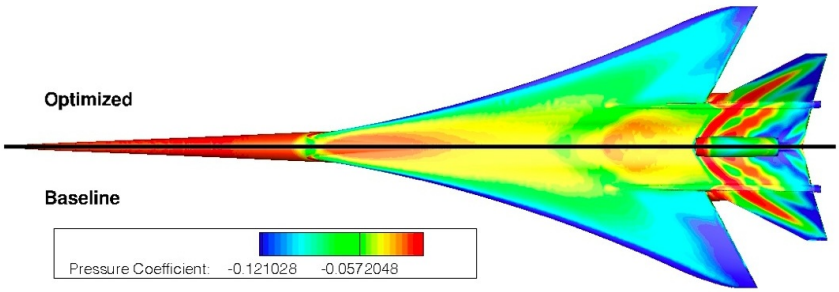
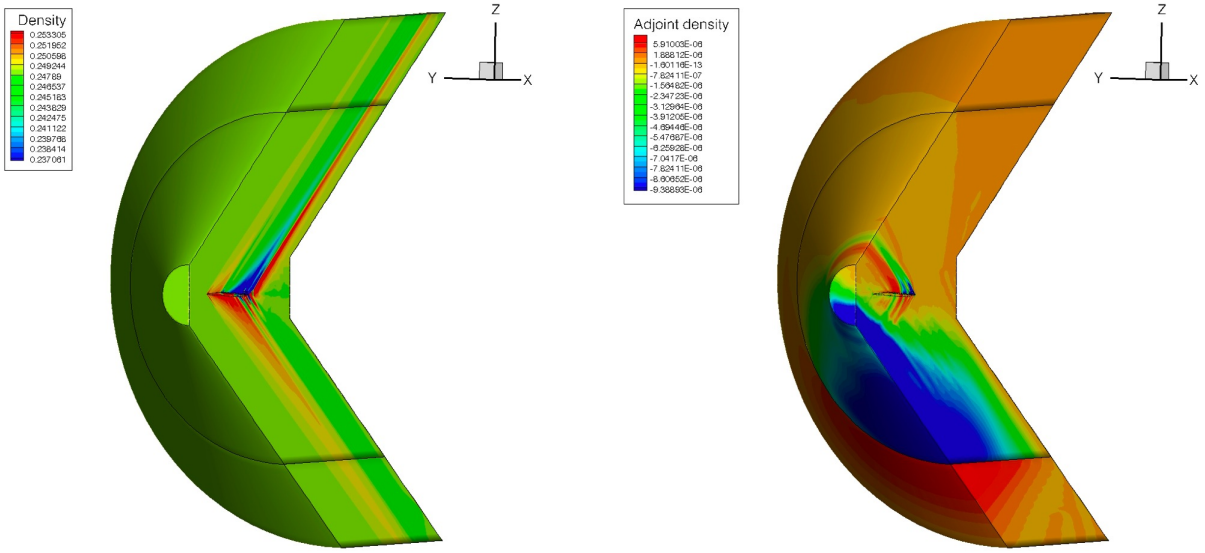


Figure 23: Baseline and Final N+2 Surface Pressures



(a) Density Field

(b) Adjoint Density Field

Figure 24: Contour Plots of Density and Adjoint Density, N+2 Baseline Configuration

IV. Conclusions

In this paper we have outlined an approach to surrogate based optimization with nonlinear constraints that addresses difficulties with hyperparameter selection and adaptive refinement. An approach to hyperparameter selection using carefully chosen constraints was used to maintain a suitable fit, and a two-stage adaptive sampling technique was developed to improve the balance of global optimization and refinement of the final optimum. The surrogate based optimizer was applied to several analytical and CFD-based test cases that identified potential improvements in optimization time. We have also introduced a new formulation of a continuous adjoint for target equivalent area distributions of multiple azimuths. The formulation was demonstrated on a combined-objective gradient-based optimization to minimize drag and maintain a target equivalent area distribution. In our future work, we expect to continue improving the surrogate based optimization methodology to handle multiple constraints, as well as applying it to the N+2 Supersonics Project.

V. Acknowledgments

The authors would like to thank the inputs from and useful discussions with Nicole Norstrud, John Morgenstern, and Michael Bounanno of the Lockheed-Martin Corporation and the financial support of the NASA Supersonics Project of the NASA Fundamental Aeronautics Program.

References

[1]Alonso, J. J. and Colonno, M., "Multidisciplinary Optimization with Applications to Sonic-Boom Minimization," *Annual Review of Fluid Mechanics*, Vol. 44, 2012, pp. 505–526.

[2]Palacios, F., Alonso, J. J., Colonno, M., Hicken, J., and Lukaczyk, T., "Adjoint-based Method for Supersonic Aircraft Design Using Equivalent Area Distributions," *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, AIAA Paper 2012-0269, Nashville, TN, January 2012.

[3]Seebass, R., "Sonic Boom Theory," *Journal of Aircraft*, Vol. 6, 1969, pp. 177–184.

[4]Chung, H. and Alonso, J., "Design of a Low-Boom Supersonic Business Jet Using Cokriging Approximation Models," *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA Paper 2002-5598, Atlanta, GA, September 2002.

[5]Myers, R. H., Montgomery, D. C., and Anderson-Cook, C. M., *Response surface methodology: process and product optimization using designed experiments*, John Wiley & Sons, Hoboke, NJ, 2009.

[6]Journel, A. G. and Huijbregts, C., *Mining Geostatistics*, Academic Press, London, 1981.

[7]Rumpfkeil, M. P., Yamazaki, W., and Mavriplis, D. J., "A Dynamic Sampling Method for Kriging and Cokriging Surrogate Models," *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, AIAA Paper 2011-0883, Orlando, FL, January 2011.

[8]Rumpfkeil, M. P., "Optimization Under Uncertainty Using Gradients, Hessians, and Surrogate Models," *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, AIAA Paper 2012-0149, Nashville, TN, January 2012.

[9]Solak, E., Murray-Smith, R., Leithead, W. E., Leith, D. J., and Rasmussen, C. E., "Derivative Observations in Gaussian Process Models of Dynamic Systems," *Advances in Neural Information Processing Systems 15*, edited by S. Becker, S. Thrun, and K. Obermayer, MIT Press, Cambridge, MA, 2003, pp. 1033–1040.

[10]Sacks, J., Welch, W., Mitchell, T., and Wynn, H., "Design and Analysis of Computer Experiments," *Statistical Science*, Vol. 4, The Institute of Mathematical Statistics, 1989, pp. 409–423.

[11]Yamazaki, W. and Mavriplis, D., "Derivative-Enhanced Variable Fidelity Surrogate Modeling for Aerodynamic Functions," *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 2011.

[12]Rasmussen, C. and Williams, C., *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006, pp. 13–30.

[13]Parzen, E., "A new approach to the synthesis of optimal smoothing and prediction systems," *Mathematical Optimization Techniques*, edited by R. Bellman, University of California Press, Berkley, CA, 1963, pp. 74–108.

[14]Paciorek, C., *Nonstationary Gaussian Processes for Regression and Spatial Modelling*, Ph.D. thesis, Carnegie Mellon University, May 2003.

[15]Turner, S., Ball, T., and Marshall, D. D., "Gaussian Process Metamodeling Applied to a Circulation Control Wing," *38th Fluid Dynamics Conference and Exhibit*, AIAA Paper 2008-4162, Seattle, Washington, June 2008.

[16]Wintzer, M. and Sturdza, P., "Conceptual Design of Conventional and Oblique Wing Configurations for Small Supersonic Aircraft," *44th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper 2006-0930, Reno, NV, January 2006.

[17]Ordaz, I. and Rallabhandi, S. K., "Boom Minimization Framework for Supersonic Aircraft Using CFD Analysis," *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, AIAA Paper 2010-1506, Orlando, FL, January 2010.

[18]Hahn, A., "Vehicle Sketch Pad: A Parametric Geometry Modeler for Conceptual Aircraft Design," *48th AIAA Aerospace Sciences Meeting*, AIAA Paper 2010-657, Orlando, Florida, January 2010.

[19]Hollingsworth, P. M. and Mavris, D. N., "Gaussian Process Meta-Modeling: Comparison of Gaussian Process Training Methods," *AIAA's 3rd Annual Aviation Technology, Integration, and Operations (ATIO) Tech*, AIAA Paper 2003-6761, Denver, Colorado, November 2003.

[20]Forrester, A. I. J. and Keaney, A. J., "Recent advances in surrogate-based optimization," *Progress in Aerospace Science*, Vol. 45, January 2009, pp. 50–79.

[21]Davis, G. J. and Morris, M. D., "Six Factors Which Affect the Condition Number of Matrices Associated with Kriging," *Mathematical Geology*, Vol. 29, 1997, pp. 669–683.

[22]Giles, M. B. and Duta, M. C., "Algorithm Developments for Discrete Adjoint Methods," *AIAA Journal*, Vol. 41, February 2003, pp. 198–205.

[23]Shimoyama, K., Sato, K., Jeong, S., and Obayashi, S., "Comparison of the Criteria for Updating Kriging Response Surface Models in Multi-Objective Optimization," *WCCI 2012 IEEE World Congress on Computational Intelligence*, 2012.

[24]Seo, S., Wallat, M., Graepel, T., and Obermayer, K., "Gaussian Process Regression: Active Data Selection and Test Point Rejection," *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, Vol. 3, Como, Italy, July 2000, pp. 241 – 246.

[25]Chung, H., *Multidisciplinary Design Optimization of Supersonic Business Jets Using Approximation Model-Based Genetic Algorithms*, Ph.D. thesis, Stanford University, March 2004.

[26]Papoulis, A., *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, New York, 1991, p. 310.

[27]Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, Vol. 13, 1998, pp. 455–492.