Reliability-Based Design of Thermal Protection Systems with Support Vector Machines

Laura M. White,* Thomas K. West IV†
and Andrew J. Brune‡
NASA Langley Research Center, Hampton, VA 23681

The primary objective of this work was to develop a computationally efficient and accurate approach to reliability analysis of thermal protection systems using support vector machines. An adaptive sampling approach was introduced informs a iterative support vector machine approximation of the limit state function used for measuring reliability. The proposed sampling approach efficient adds samples along the limit state function until the reliability approximation is converged. This methodology is applied to two sample, mathematical functions to test and demonstrate the applicability. Then, the adaptive sampling-based support vector machine approach is applied to the reliability analysis of a thermal protection system. The results of all three problems highlight the potential capability of the new approach in terms of accuracy and computational saving in determining thermal protection system reliability.

Nomenclature

b	Bias	$\mathbf{U}$	Vector of random variables in U space
C	Misclassification cost	$\mathbf{w}$	Normal vector to hyperplane
d_i	Distance between samples	$\mathbf{x}$	Data point
d_{final}	dynamic distance	$\mathbf{X}$	Vector of random variables in x space
$f(\mathbf{x})$	Probability density function (PDF)	X	Sample space
$F(X)$	Cumulative distribution function	X_{close}	Close data points to SVM
$g(\mathbf{U})$	Limit-state function in U space	X^+	Data points with class $+1$
$g(\mathbf{X})$	Limit-state function in X space	X^-	Data points with class -1
$I_k(\mathbf{x})$	Indicator function	X^+_{close}	Close data points to SVM with class $+1$
k	Algorithm iteration	X^-_{close}	Close data points to SVM with class -1
K	Kernel function	$X_{\text{constructive}}$	Constructive data set for SVM
n	Dimension of space	y	data classification
N	Number of training samples	ξ	Slack variable
N_{run}	Max number of samples per iteration	Φ	Standard normal distribution
N_{test}	Number of testing points for convergence	Δ_k	Fraction of convergent points
p_f	Probability of failure	$\widehat{\Delta}$	Fitted exponential to Δ_k values
r	Line connecting data of opposite signs	ϵ_1	Stopping criterion for Δ_k and $\widehat{\Delta}_k$
$s(\mathbf{x})$	SVM generated from X	ϵ_2	Stopping criterion of slope of $\widehat{\Delta}$
$\mathbf{u}^*$	Most probable point		

I. Introduction

Thermal protection systems (TPS) are crucial to planetary vehicle design due to the high aerodynamic heating that is caused when entering the atmosphere. The TPS protects the spacecraft by acting as a shield

*Mathematician, Vehicle Analysis Branch, Systems Analysis and Concepts Directorate.
†Aerospace Engineer, Vehicle Analysis Branch, Systems Analysis and Concepts Directorate, Member AIAA.
‡Aerospace Engineer, Structural and Thermal System Branch, Engineering Directorate, Member AIAA

against heating caused by compression, surface friction, and radiation of atmospheric gas. Different types of designs include ablative, thermal soak, and hot structure heat shields. A crucial component to TPS design is the sizing of the material to adequately protect the spacecraft and to ensure the safety of any on-board cargo and crew.

Having exact knowledge of how to size a TPS is difficult due to the large amount of uncertainty present as a result of complex flow physics, material properties, and modeling approaches. To combat this, uncertainty quantification (UQ) is performed on the heat shield to determine the expected performance of a TPS as a result of all known uncertainties. This can then be used to estimate the reliability of the TPS in satisfying the mission requirements.

One challenge in quantifying the reliability of a TPS is the lack of an explicit, functional form of the limit state which defines the separation between the failure zone and safe zone. Instead, an implicit limit state function must be used to perform a reliability analysis. Implicit limit state functions cause difficulties in gradient-based reliability methods,[1] such as the first-order reliability method (FORM). A simulation-based method, such as a Monte Carlo simulation (MCS), can be used, but with high computational cost. Global regression of the domain can also be performed with a surrogate model, but requires an accurate representation of the domain spanned by the uncertain parameters. This may be difficult if the domain is not smooth or is highly nonlinear. To mitigate the lack of knowledge of the limit state function, previous work has been introduced to use support vector machines (SVM) to approximate the limit state function.[2,3] With the support vector machine approximation, gradient-based reliability methods, such as FORM, can be used to approximate the failure probability of the TPS design. However, this approach adds challenges such as selecting the size of the initial data needed for an accurate construction of an approximate limit state, which may be large.

In literature, there have been several different proposed methods that reduce the number of samples by selectively generating new samples. The method of explicit design space decomposition (EDSD) was developed for SVMs and applied to several application problems.[3,4] This strategy adds new training data on the boundary of the SVM by solving a global optimization problem using a genetic algorithm (GA). This proposed method handles discontinuous and disjoint problems successfully. Another approach uses a virtual support vector machine (VSVM), where virtual samples are generated near the limit state function by using a Kriging model to approximate the limit state.[5] This method successfully handles continuous problems while adding a small number of virtual sample points, leading to a small number of function calls. The objective of this paper is to expand on previous work and introduce an algorithm to update an approximate support vector machine based limit state function from a small initial data set by using an adaptive sampling strategy that focuses on producing an accurate approximate limit state based upon previous sample points and model response information. The goal is then to demonstrate, not only the new technique, but also a computationally efficient and accurate approach to design under uncertainty from the standpoint of reliability of thermal protection systems.

Basic concepts and important features of SVM are presented in Section II.A. Section II.B introduces how to find the reliability using FORM. The adaptive sampling strategy including the convergence criterion are presented in Section II.C, followed by examples problems given in Section II.D. In Section III, the application problem to reliability of TPS will be presented based on the methods that are proposed in this paper. Conclusions will be given in Section IV.

II. Reliability-Based Design with Machine Learning

Reliability analysis is an important component to reliability-based optimization (RBDO). The status of success is given by a limit state function, which is derived from physical properties.[6] Typically, limit state functions are denoted by $g(\mathbf{X})$, where $\mathbf{X}$ is a random vector of length n. An event is successful when $g(\mathbf{X}) \geq 0$ and a failure event when $g(\mathbf{X}) < 0$. The probability of failure then becomes

$$p_f = \Pr\{g(\mathbf{X}) < 0\} = \int_{g(\mathbf{X})<0} f(\mathbf{x}) d\mathbf{x} \tag{1}$$

where $f(\mathbf{x})$ is the joint probability density function (PDF) of $\mathbf{X}$. However, the integral in Eq. (1) is difficult to compute as the limit state is typically not explicitly defined. This section covers introductory material and discusses an approach to evaluating this integral. Approximating the limit state function using SVM

with an adaptive sampling strategy is also presented in detail. Example problems are then given to illustrate the process and applicability.

A. First-Order Reliability Method and the Most Probable Point

One approach to evaluating the integral in Eq. (1) is to use an approximation method called FORM.[7] FORM linearizes the limit state at the highest probability point producing the value of the limit state function, called the Most Probable Point (MPP). Assuming that all random variables in $\mathbf{X}$ are independent, FORM is broken down into three steps.[8,9]

1. Transform random variables from X-space into standard normal variables in U-space. The original random variables $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ are transformed into $\mathbf{U} = (U_1, U_2, \ldots, U_n)$ by $U_i = \Phi^{-1}[F_i(X_i)]$ for $1 \leq i \leq n$ where $F_i(X_i)$ is the cumulative distribution function (CDF) of X_i and Φ^{-1} is the inverse CDF of a standard normal variable.

2. Perform a search for the MPP, which is found on the limit state $g(\mathbf{U}) = 0$. The MPP will be the point that maximizes the joint PDF of $\mathbf{U}$. This leads to the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{U}} \|\mathbf{U}\|, \\ &\text{subject to } g(\mathbf{U}) = 0 \end{aligned} \tag{2}$$

   Solving this, the MPP $\mathbf{u}^* = (u_1^*, u_2^*, \ldots, u_n^*)$ is obtained. To simplify calculations the integration boundary $g(\mathbf{U}) = 0$ is approximated by a first order Taylor expansion, as shown in Eq. (3).

$$g(\mathbf{U}) \approx g(\mathbf{u}^*) + \nabla g(\mathbf{u}^*)(\mathbf{U} - \mathbf{u}^*)^T \tag{3}$$

   Here, T stands for the transpose and $\nabla g(\mathbf{u}^*)$ is the gradient of g at the MPP. This gives rise to the reliability index $\beta = \|\mathbf{u}^*\|$, which will be minimum distance from the origin to the MPP. Note that the use of higher order terms is possible but requires the calculation of higher order derivatives.

3. Calculate the probability of failure at the reliability index found in previous step. This is computed by

$$p_f = P\{g(\mathbf{X}) < 0\} = \Phi(-\beta) \tag{4}$$

   where, Φ is the CDF of a standard normal variable.

One of the advantages to FORM is that the transformation performed in step 1, called the Rosenblatt Transformation,[10] can be used on normal and non-normal distribution of each X_i. Although, FORM is very useful and easy to implement, it does have several challenges. One is that FORM has difficulty converging when applied to implicitly defined limit state equations, which is often the case for many engineering applications. Any numerical error in calculating the function value or gradients can cause the MPP search to not converge to a point. One approach to combat this is to construct a surrogate of the limit state or the boundary between safe and fail zones. Many techniques exist for surrogate modeling, such as polynomial chaos expansions and Kriging models. Surrogate modeling is often plagued by the curse of dimensionality and may be challenging to fit highly nonlinear limit state functions. The use of support vector machines can mitigate these challenges with regression-based approaches.

B. Support Vector Machines

Support vector machines are a machine learning technique that is becoming increasingly popular for classification of nonlinear data.[11,12] The algorithm was originally introduced by Vladimir Vapik for linear classification of data and later extended to nonlinear classifiers by applying a kernel trick, which maximizes the margins in a hyperplane.[13] One of the main features of SVM is that it gives the ability to define a decision function that optimally separates two classes of data samples.

Given a set $X \subseteq \mathbb{R}^n$ of N training samples x_i in an $n-$dimensional space, each sample has a class associated with it, characterized by the value $y = \pm 1$. For linear data, the SVM algorithm finds the decision boundary that maximizes the margin between two parallel supporting hyperplanes that separate the data. The margins are produced such that there is at least one data point that lies on the margin and no data

in between the two margins. The points that lie on the hyperplanes are called the support vectors and the decision boundary will lie halfway between the two hyperplanes.

This formulation leads to the following mathematical setup.[14] A given data point $\mathbf{x} = (x_1, \ldots, x_n) \in X$ is assigned to the positive class if data lies in one type of classification and negative class if data lies in the other class. The two parallel hyperplanes that define the margin are described by

$$\mathbf{w} \cdot \mathbf{x} + b = 1 \tag{5}$$

and

$$\mathbf{w} \cdot \mathbf{x} + b = -1 \tag{6}$$

where $\mathbf{w}$ is the normal vector to the hyperplane and b is the bias. The maximum-margin hyperplane between the two classifications will be characterized by

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{7}$$

which will be halfway between the two support hyperplanes as illustrated in Figure 1. All points associated with $y = +1$ lead to a positive value of the SVM and points associated with $y = -1$ are negative values of the SVM. Eqs. (5) and (6) along with the constraint that no point can lie in between the support hyperplanes becomes the constraint problem

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \tag{8}$$

where $\mathbf{x}_i \in X$ and $1 \leq i \leq N$. The distance between the two supporting hyperplane is $\frac{2}{\|\mathbf{w}\|}$. Thus, in order to maximize the distance, $\|\mathbf{w}\|$ must be minimized. Therefore, the following optimization problem is produced

$$\begin{aligned} &\min_{\mathbf{w},b} \ \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0, \quad 1 \leq i \leq N \end{aligned} \tag{9}$$

where $\mathbf{x}_i \in X$.

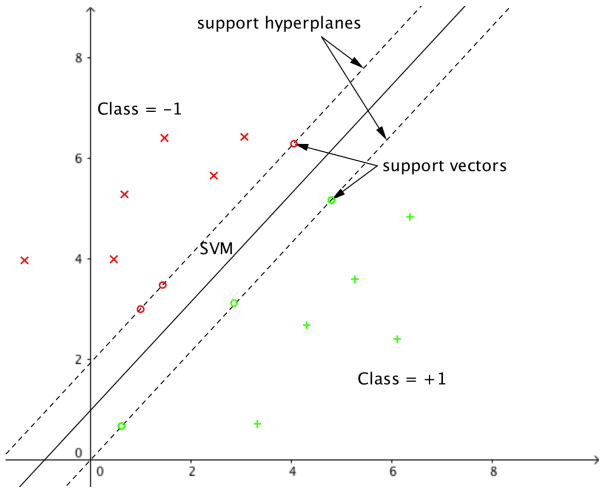


Figure 1: Optimal separating hyperplane in two-dimensional linear problem

To solve the problem in Eq. (9), it is transformed into its corresponding quadratic optimization problem. The result then becomes

$$\begin{aligned} &\max_{\alpha} \ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \mathbf{x}_i \cdot \mathbf{x}_j \\ &\text{subject to } \sum_{i=1}^{N} \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad 1 \leq i \leq N \end{aligned} \tag{10}$$

where α_i are the Lagrangian multipliers. Solving (10) gives the optimal Lagrange multipliers α_i and the associated hyperplane normal vector, $\mathbf{w}$, and bias value, b. These values can then be used to determine the

classification of any test point $\mathbf{x} \in X$, which is given by Eq. (11).

$$s(\mathbf{x}) = b + \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x}. \tag{11}$$

The Kuhn and Tucker conditions of the optimization problems gives that only the Lagrange multipliers associated with the support vectors will be nonzero. Eq. (11) then becomes

$$s(\mathbf{x}) = b + \sum_{i=1}^{\text{NSV}} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} \tag{12}$$

where NSV is the number of support vectors.

For data that is not linearly separable, the optimization problem in Eq. (9) is not possible and the inequality constraints are relaxed by using a non-negative slack variable ξ_i. The optimization problem becomes

$$\begin{gathered} \min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi \\ \text{subject to } y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 \geq \xi_i, \quad 1 \leq i \leq N \end{gathered} \tag{13}$$

where the coefficient C is called the misclassification cost and becomes the upper bound for the Lagrange multipliers in the dual formulation.

SVMs can be extended to nonlinear decision functions by projecting the samples into a higher dimensional space, called feature space. There is a class of functions $K(\mathbf{x}_i, \mathbf{x}_j)$ with the following property: there exist a linear space F and a mapping ϕ from X to F such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{14}$$

where $\langle \cdot, \cdot \rangle$ is the inner product in F. Often, K is referred to as the kernel. Using this formulation, the classification for nonlinear data is obtained by the sign of

$$g(\mathbf{x}) = b + \sum_{i=1}^{\text{NSV}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) \tag{15}$$

where K is the kernel, $\mathbf{x}_i \in X$, $y_i \in \{+1, -1\}$, α_i are the Lagrangian multipliers, and b is the bias.

The two most commonly used kernel functions are the Gaussian kernel and polynomial kernels. The Gaussian kernel is defined as

$$K(\mathbf{x}_i, \mathbf{x}) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2}} \tag{16}$$

where σ is the width factor. The polynomial kernel is defined as

$$K(\mathbf{x}_i, \mathbf{x}) = (1 + \mathbf{x}_i'\mathbf{x})^d \tag{17}$$

where $d \in \mathbb{N}$. Choosing a kernel is highly problem dependent and still needs to be analyzed to determine the most appropriate kernel given a data set.

C. Adaptive Sampling

Typically, the number of training samples needed for an SVM to construct an accurate limit state function depends on the complexity of the function. In general, a very large random sample must initially be generated to make a good prediction of the limit state function. This requires a high number of function calls, which leads to computationally inefficient algorithms. To combat this issue, this paper proposes an adaptive sampling strategy, which places new samples close to the limit state function. The algorithm introduced here uses similar strategy to the work of Song et al.[5] but without the use of a kriging model and applies a different adaptive sequential sampling routine.

An SVM is constructed from initial samples that are generated using a Latinized Centroidal Voronoi Tessellation Design of Experiments (LCVT DOE), which has the ability to produce good uniformity and

randomness in the sample space.[3,15] Each sample point has a classification associated with it, i.e., success (+1) or failure (−1), and the true limit state function is located between the classifications. The proposed algorithm here uses a sequential sampling strategy that applies the opposite signed samples to generate new samples that are close to the SVM but are far away from samples in the initial set. The sequential sampling strategy approach can be seen in Figure 3. Adding samples using this method allows for gaps in the sample space to be filled which in return gives a more informative SVM with a better approximation of the actual limit state. The remainder of this section will introduce the selection of the SVM in Section C.1, the sequential sampling strategy in Sections C.2 and C.3, and the convergence criterion in Section C.4.

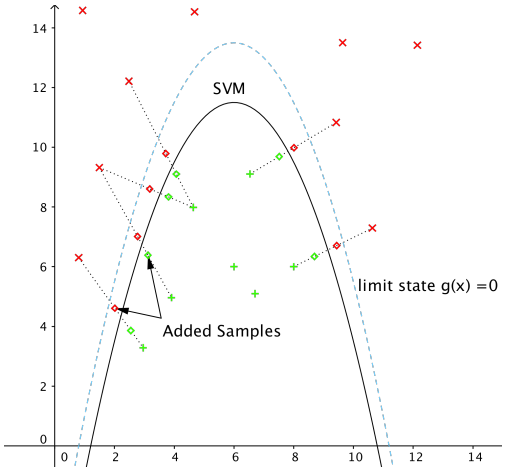


Figure 2: The samples that are added to the constructive set, $X_{\mathbf{constuctive}}$

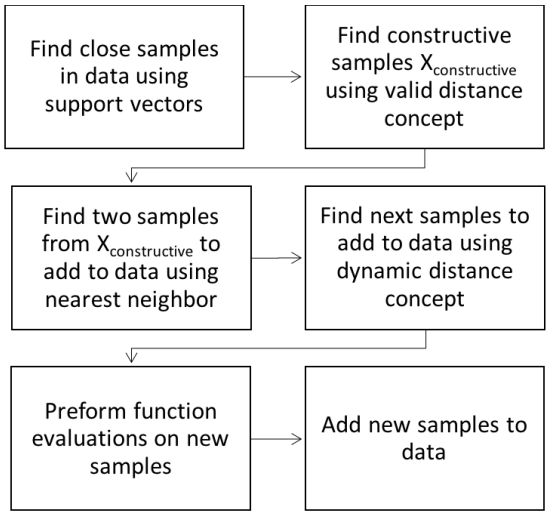


Figure 3: Flowchart for sequential sampling strategy

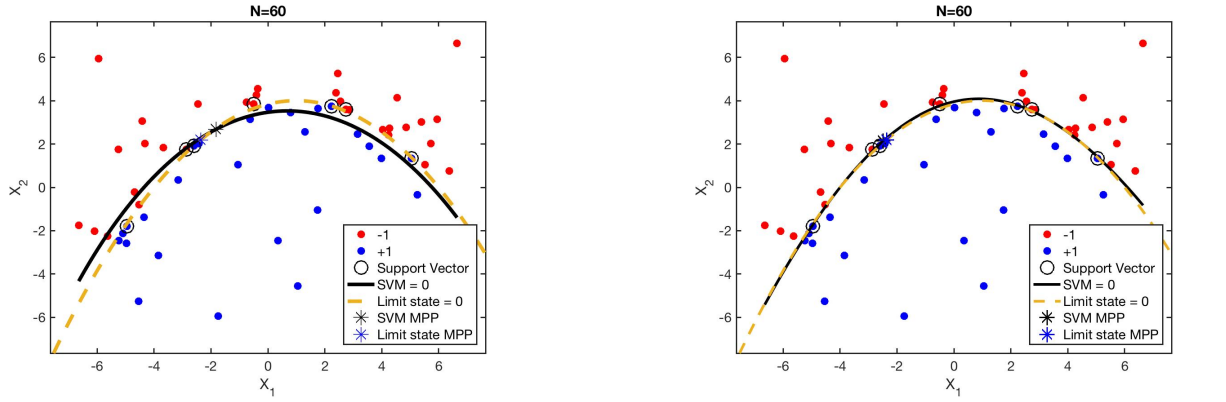
C.1. Selection of the Support Vector Machine

Depending on the chosen kernel, there are several parameters within the selection of the SVM that can hinder how the SVM behaves. Given that there is initially low knowledge of the problem at hand, a cross-validation procedure is introduced to determine the unknown parameters. Cross-validation gives a prediction for the fit of a model to a test validation set when an explicit validation set is not available. If these parameters aren't selected properly when data points are close to each other there is higher likelihood of over fitting. The way the adaptive sampling routine works here is that it adds success and fail samples around the SVM which leave data points near each other when convergence is close. Therefore, it is particularly important to have appropriate parameters for the kernel for this routine. The work here focuses on cross-validation procedure

for the polynomial kernel, which has unknown parameters: box constraint, kernel scale, and degree. The cross-validation procedure here is only performed on the box constraint and kernel scale due to the rapid fluctuations in the SVMs when the degree is changed. A visual comparison between the converged program when the SVM has and has not been cross-validated can be seen in Figure 4.

The cross-validation model is executed through a k-fold cross-validation procedure.[16] This method is performed by randomly partitioning the original sample into k equal sized subsamples. Then, $k-1$ subsamples are used as training data, and the single remaining sample is used for the validation data. A single estimation is then produced using the average of the k results from the folds.

The SVM is created using MATLAB's SVM function fitcsvm with the parameter to optimize hyper-parameters (kernel scale and box constraint) turned on. The default settings are used, which applies a 5-fold cross-validation model to validate an appropriate scale and box constraint for the model. The SVM is constructed this way for each iteration. Changing parameters each iteration the convergence criterion will have rapid fluctuations in the beginning, but as more samples are added the fluctuations start to decrease. Although not used here, the cross-validation model could potentially also be used to determine the degree of the polynomial and will be investigated in future work.



(a) Not cross-validated SVM, Box Constraint = 1, Kernel Scale = 1

(b) Cross-validated SVM, Box Constraint = 930.96, Kernel Scale = 0.0863

Figure 4: A visual representation of over fitting of the SVM when the model is not cross-validated for the converged adaptive sampling algorithm.

C.2. Valid Distance and Constructive Samples

The procedure to generate the constructive samples by applying the valid distance concept was originally introduced in Song et al.[5] The process starts out with a set of samples that have produced limit state evaluations, called X. Given that set, close samples are determined and added to a new set, denoted X_{close}. To find which samples will be added to X_{close}, the limit state evaluation is considered. Samples that have a small limit state evaluation should be close to the actual limit state. Also, support vectors should lie close to the actual limit state. Therefore, the close samples are samples whose limit state evaluation are within the largest support vector limit state evaluation. This can be described as

$$\begin{aligned} g_{\max} &= \max_i(|g(\mathbf{x}_i^*|),\ i = 1, \ldots, N_{\text{SV}} \\ X_{\text{close}} &= \{\mathbf{x} \in X \mid |g(\mathbf{x})| \leq g_{\max}\} \end{aligned} \tag{18}$$

where $\mathbf{x}_i^*$ is the ith support vector, N_{SV} is the number of support vectors, and g is the limit state evaluation at the given sample.

Next, the constructive set, $X_{\text{constructive}}$, is created using a valid distance criterion.[5] The valid distance is the largest distance among all distances between the closest pairs. The criterion can be established by first separating X_{close} in two sets, one that consists of the success samples according to the SVM (not the actual limit state), called X_{close}^+ and the failure samples according to the SVM, called X_{close}^-. Then the distances

between all of the samples in X^+_{close} and X^-_{close} is calculated by

$$\begin{aligned} D_i &= \{d_1, \cdots, d_{N^-_{\text{close}}}\},\ i = 1, \ldots, N^+_{\text{close}} \\ d_j &= \|\mathbf{x}^+_i - \mathbf{x}^-_j\|,\ j = 1, \ldots, N^-_{\text{close}} \end{aligned} \tag{19}$$

where N^+_{close} is the number of samples in X^+_{close}, $\mathbf{x}^- \in X^-_{\text{close}}$, and $\mathbf{x}^+ \in X^+_{\text{close}}$. Then, the valid distance will be maximum distance of the smallest distance between the opposite signed samples. Mathematically, it becomes

$$\text{valid distance} = \max_i \min_{d_j \in D_i} D_i \tag{20}$$

for $i = 1, \ldots, N^+_{\text{close}}$. Then, given a failure sample $\mathbf{x}^- \in X^-_{\text{close}}$ and a success sample $\mathbf{x}^+ \in X^+_{\text{close}}$ which satisfies the valid distance criterion a line is drawn between them by

$$\mathbf{r} = \mathbf{x}^+ t + \mathbf{x}^-(1 - t) \tag{21}$$

where $t \in [0, 1]$. Two samples are then added to $X_{\text{constructive}}$ by moving $\mathbf{x}^+$ along the line $\mathbf{r}$ until it reaches the failure side of the SVM and $\mathbf{x}^-$ until it reaches the success side of the SVM. This is repeated with all opposite signed samples within X_{close} that satisfy the valid distance criterion. The final product is the constructive sample set, $X_{\text{constructive}}$, which can be seen in Figure 2.

C.3. Sample Selection and Dynamic Distance

The $X_{\text{constructive}}$ set typically has a lot of samples that are close to each other and if were added to the design space could cause over fitting of the SVM. Therefore, the next step is to develop an effective way to find samples in $X_{\text{constructive}}$ which are informative. The informative samples should be far away from any sample that is already in the design space and any sample that will be added to the design space. This is done by first establishing an initial success sample and failure sample in $X_{\text{constructive}}$ that is far away from samples in X. After the two initial samples have been established, the next samples are added such that they are far away from the initial samples and also samples in X. The remainder of this section describes how these samples are picked in detail.

First a success sample and failure sample is added by finding which ones have the largest distance from samples in X. The search for this is done by the nearest neighbor function in MATLAB's statistics and machine learning toolbox.[17] The nearest neighbor algorithm creates an efficient and effective way to see which samples in X are closest to the samples in $X_{\text{constructive}}$.[18] Once a nearest neighbor search is done for all samples in $X_{\text{constructive}}$, the first pair is chosen by the maximum distance of the success and failure sample from the results given by the nearest neighbor search, denoted $\text{knn}(\cdot, \cdot)$. An illustration of the initial samples can be seen in Figure 5. To phrase this mathematically, take $X^+_{\text{constructive}}$ and $X^-_{\text{constructive}}$ to be the set of samples with success (i.e. of class $+1$) and failure (i.e. of class -1) for the SVM in $X_{\text{constructive}}$, respectively. Then the distances of the nearest neighbors for success and failure samples to all samples in X are given by

$$d^+_i = \text{knn}(\mathbf{x}^+_i, X) \text{ and } d^-_i = \text{knn}(\mathbf{x}^-_i, X),\ i = 1, \ldots, N_{\text{pairs}} \tag{22}$$

where $\mathbf{x}^-_i \in X^-_{\text{constructive}}$, $\mathbf{x}^+_i \in X^+_{\text{constructive}}$, and N_{pairs} are the number of pairs added to create $X_{\text{constructive}}$. Now the constructive samples chosen to be added to X is will be the maximum nearest neighbor for each success and failure sample, which becomes

$$\mathbf{x}^+ = \max_i(d^+_i) \text{ and } \mathbf{x}^- = \max_i(d^-_i). \tag{23}$$

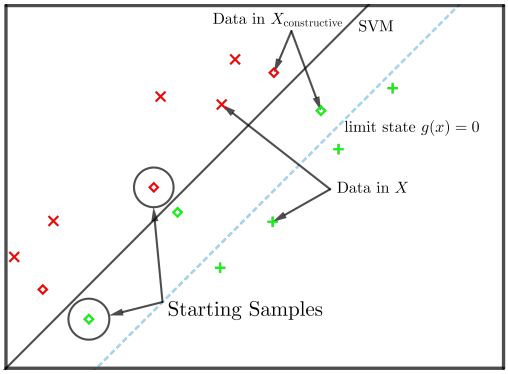


Figure 5: Illustration of how the first two samples are selected for function evaluation.

After the constructive samples have been chosen, the next samples are chosen based on those samples and the dynamic distance. The next sample should be far away from the previously chosen sample and as well as the data that is already in X. How far away the sample should be is decided by the dynamic distance. The distance is based on the maximum number of samples that is allowed for each run, called N_{run}. This sample size can be user defined, i.e. if the user wishes to add a max of 10 samples each run then it is possible. The distance is chosen by an iteration method of determining how many samples are added per a varying distance parameter. The distance parameter that gives the sample size below the desired maximum samples is chosen as the dynamic distance, denoted by d_{final}.

The distance parameters are generated from a logarithmic function. A logarithmic function is increasing, but at a slow pace, which allows for a wide selection of slowly increasing distance parameters. First, a grid of evenly spaced points is created for the logarithmic function which starts at one and ends at e^2. The grid becomes

$$M = \{m_1 = 1, m_2, m_3, \ldots, m_{299}, m_{300} = e^2\} \tag{24}$$

where each m_i is evenly spaced for $1 \leq i \leq 300$. The value e^2 is chosen as the final grid point since $\log(e^2) = 2$, which will account for all the samples in X that are close to the SVM. Also, for a small number of desired added samples looking at distances larger than the 2 will only result in adding the two initial samples to the data space. The grid is applied to the logarithmic function to create an array of distances, $d_i = \log(m_i)$ for $1 \leq i \leq 300$ and $m_i \in M$.

The sample which is the farthest distance from the previously added sample, $\mathbf{x} \in X_{\text{constructive}}$, is tested against each distance parameter, d_i. The sample $\mathbf{x}$ is tested to see if the distance from $\mathbf{x}$ and every sample in X and every sample in $X_{\text{constructive}} \backslash \{\mathbf{x}\}$ lies outside the distance d_i. If in both cases the sample lies outside the distance d_i, then it would be added to X. The problem for which each sample will be added per distance d_i can be written as

$$\mathbf{x}^i_{\text{added}} = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}_{\text{sample}}\| \geq d_i \text{ and } \|\mathbf{x} - \mathbf{x}_{\text{constructive}}\| \geq d_i,\ \mathbf{x}_{\text{sample}} \in X,\ \mathbf{x}_{\text{constructive}} \in X_{\text{constructive}} \backslash \{\mathbf{x}\}\} \tag{25}$$

for $1 \leq i \leq 300$. Therefore, when the distance is small, there will be a lot of added samples and as the distance grow, less samples will be added. Once the iteration goes over the desired number of samples to be added the iteration stops and d_{final} is chosen to be the d_i that had less samples than N_{run}. Formally, the dynamic distance can be written as

$$\begin{gathered} d_{\text{final}} = d_i \\ \text{subject to } N_i \leq N_{\text{run}} \end{gathered} \tag{26}$$

where N_i are the number of samples generated for distance parameter d_i with $1 \leq i \leq 300$. The samples that are selected to be added to X are the ones that satisfy the dynamic distance of d_{final}. With the additional samples added a new SVM is generated and to process repeats until convergence. The sequential sampling strategy is shown in Figure 3, while the overall procedure for the adaptive sampling method is shown in Figure 6.

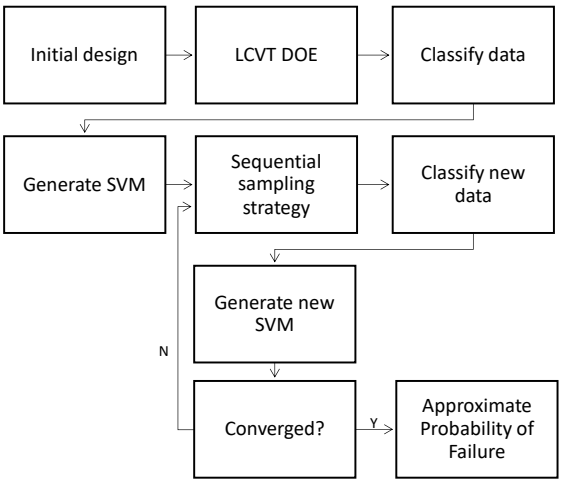


Figure 6: Flowchart for adaptive sampling algorithm

C.4. Convergence Criterion

Because the actual limit state equation is not known, the stopping criterion is based upon variation of the generated SVMs from the algorithm. A set of N_{test} testing points is generated using the appropriate distribution of the sample parameters. The fraction of convergence points for which the sign of the SVM evaluation changes between iterations[3,5] is calculated by

$$\Delta_k = \frac{\sum_{i=1}^{N_{\text{test}}} I_k(\mathbf{x}_i)}{N_{\text{test}}} \tag{27}$$

where k is the current iteration number. Take I_k to be the indicator function defined by

$$I_k(\mathbf{x}_i) = \begin{cases} 0, & \text{sign}(s_{k-1}(\mathbf{x}_i)) = \text{sign}(s_k(\mathbf{x}_i)) \\ 1, & \text{otherwise} \end{cases} \tag{28}$$

where $s_{k-1}(\mathbf{x}_i)$ and $s_k(\mathbf{x}_i)$ represents the SVM generated at $k-1$th and kth iteration, respectively. Note that N_{test} can be very large because Δ_k is only dependent about evaluations along the SVM, which is computationally inexpensive.

To implement a more stable stopping criterion Basudhar and Missoum[3] performed an exponential curve fit on the fraction of testing points change signs given by

$$\widehat{\Delta}_k = Ae^{Bk} \tag{29}$$

where $\widehat{\Delta}_k$ represents the fitted values for Δ_k and A and B are the parameters of the exponential curve. As argued by authors Song, et al.,[5] if Δ_k is large and $\widehat{\Delta}_k$ is small than there was a large change in the SVM update that the kth iteration in which $\widehat{\Delta}_k$ did not catch. Also, if Δ_k is small and $\widehat{\Delta}_k$ is large than a new sample was inserted and didn't change the SVM much, but it may not have converged yet. Hence, both Δ_k and $\widehat{\Delta}_k$ should be kept small, and the slope of the curve needs to be small for stable convergence.

Now the max between the fitted value and fraction of convergence points should be less than a small parameter ϵ_1 while the absolute value of the slope of the curve at convergence should be lower than ϵ_2. This leads to the stopping criterion

$$\max\{\Delta_k, \widehat{\Delta}_k\} < \epsilon_1 - \epsilon_2 < BAe^{Bk} < 0. \tag{30}$$

In general, ϵ_2 is smaller than ϵ_1 since the slope of the fitted curve should decrease faster as more iterations occur.

D. Example Problems

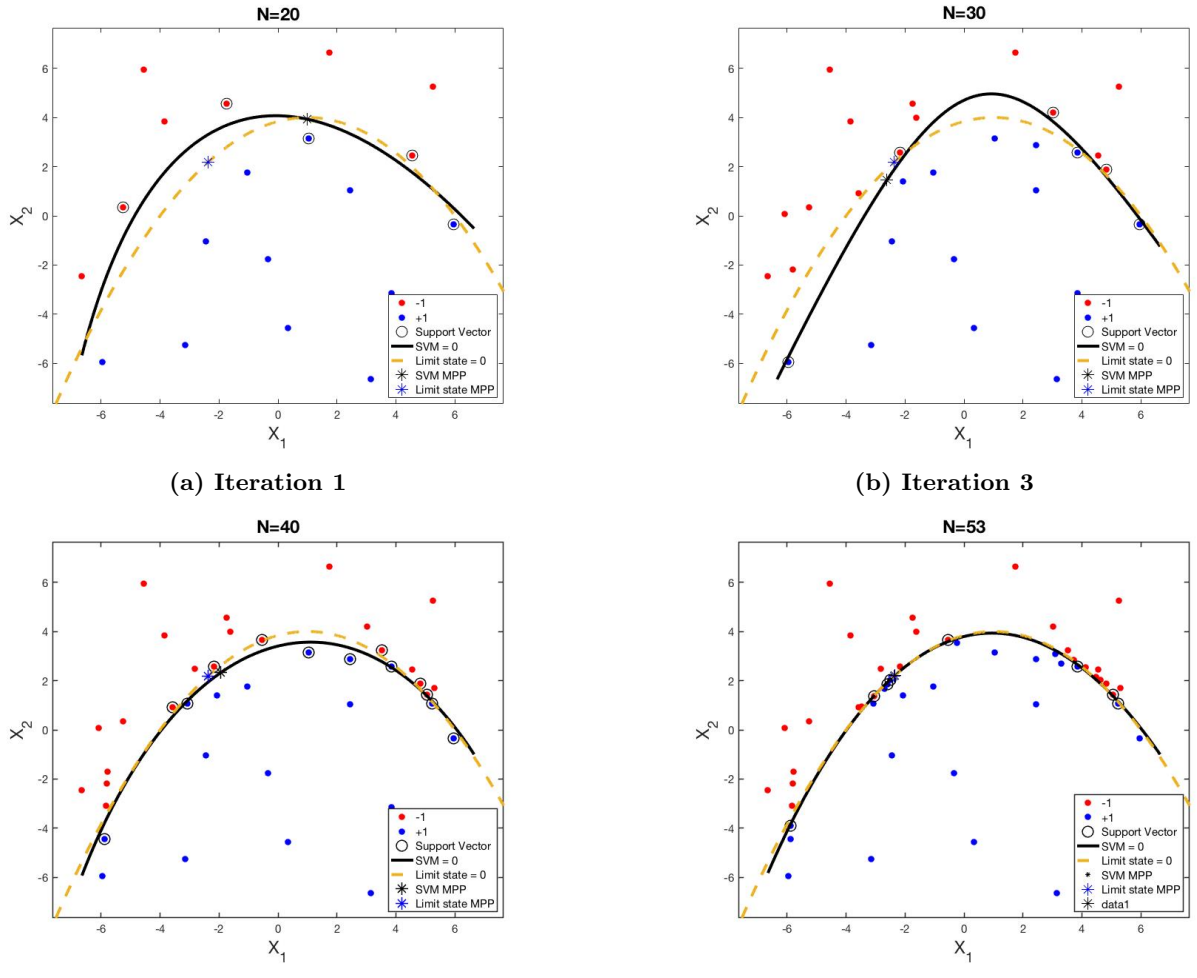
Two examples are presented here to demonstrate the accuracy and efficiency of the presented method. The MCS probability of failure for the adaptive SVM is shown in each example, due to the low-cost requirement of running a lot of samples on the SVM. Each example compares the failure probabilities using FORM and MCS to the failure probabilities using SVM-based FORM and SVM-based MCS. For all examples, the number of testing samples is given by $N_{\text{test}} = 1,000,000$.

D.1. Example 1

Consider a quadratic limit state reliability analysis[2,19] where the limit state is defined as

$$g(\mathbf{x}) = 4 - \frac{4}{25}(x_1 - 1)^2 - x_2 \tag{31}$$

where x_1 and x_2 are standard normal random variables. The initial LCVT DOE was chosen to be of size 20 that are 7σ from the mean in order to ensure failure points. The kernel was chosen to be a second order polynomial and the maximum number of added samples per iteration was $N_{\text{run}} = 5$. A total of 1,000,000 Monte Carlo samples were generated using Latin Hypercube sampling and tested against the SVM. A few of iterations are shown in Figure 7. Initially, the SVM does a poor job of fitting the actual limit state, but as informative samples are added, the SVM drastically improves. The convergence criterion was 0.03 and 0.005 for ϵ_1 and ϵ_2, respectively. The convergence can be seen visually in Figure 8, which shows the ϵ_1 versus the number of iterations and the exponential fit $\widehat{\Delta}_k$. The results are summarized by Table 1 below.



(a) Iteration 1 **(b) Iteration 3**

(c) Iteration 5 **(d) Iteration 8**

Figure 7: Example 2 - The changes in the SVM during the adaptive sampling process

Table 1: Sample size and failure probabilities for Example 2 with $\epsilon_1 = 0.03$ and $\epsilon_2 = 0.004$.

N_{initial}	N_{added}	MCS	FORM	Adaptive SVM-based	
				MCS	FORM
20	33	0.000794	0.000636	0.000808	0.000628

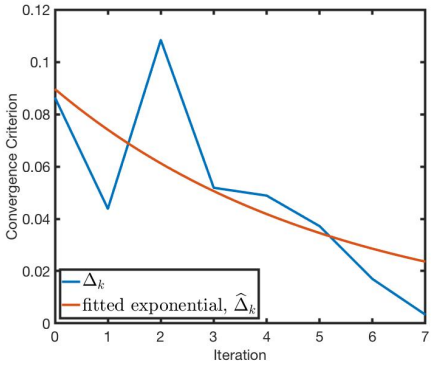


Figure 8: Example 1 - number iterations vs. convergence criterion

D.2. Example 2

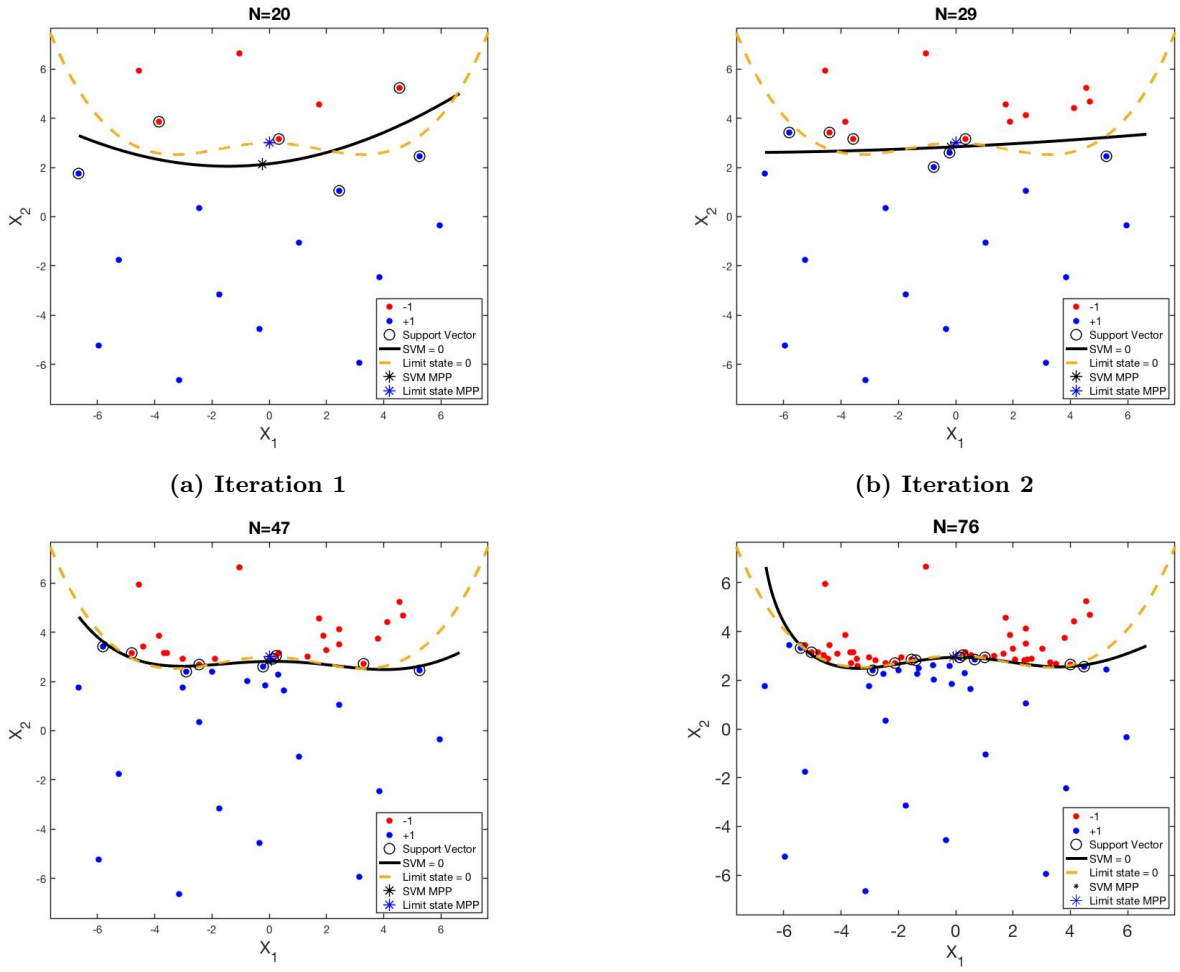
Consider a fourth order limit state reliability analysis[2,20] where the limit state is defined as

$$g(\mathbf{x}) = 2 + \exp\left(-\frac{x_1^2}{10}\right) + \left(\frac{x_1}{5}\right)^4 - x_2 \tag{32}$$

where x_1 and x_2 are standard normal random variables. The initial LCVT DOE was chosen to be of size 20 that are 7σ from the mean in order to ensure failure points. The kernel was chosen to be a fourth order polynomial and the maximum number of samples per iteration was $N_{\text{run}} = 10$. A few of the iterations are shown in Figure 9. As before, initially the SVM does not fit the actual limit state very well, but as informative samples are added, the SVM improves. The convergence criterion was 0.03 and 0.004 for ϵ_1 and ϵ_2, respectively. The convergence can be seen visually in Figure 10 which shows the ϵ_1 versus the number of iterations and the exponential fit $\widehat{\Delta}_k$. The results are summarized by Table 2.

Table 2: Sample size and failure probabilities for Example 2 with $\epsilon_1 = 0.03$ and $\epsilon_2 = 0.004$.

N_{initial}	N_{added}	MCS	FORM	Adaptive SVM-based	
				MCS	FORM
20	56	0.00184	0.00135	0.00207	0.00165



(a) Iteration 1 **(b) Iteration 2**

(c) Iteration 4 **(d) Iteration 7**

Figure 9: Example 2 - The changes in the SVM during the adaptive sampling process

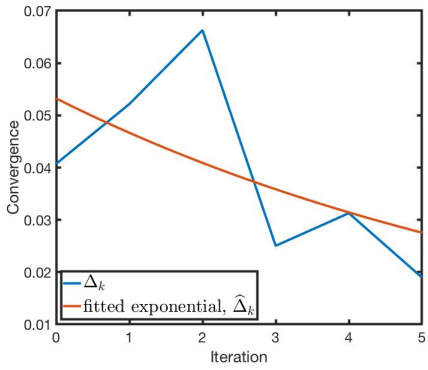


Figure 10: Example 2 - number iterations vs. convergence criterion

III. Thermal Protection System Reliability Analysis

In this section, the SVM technique presented in the previous section will be demonstrated on the reliability analysis of a flexible TPS system. The TPS layup considered consists of two layers of outer fabric material, four layers of insulation, and an impermeable gas barrier. A one-dimensional physics-based thermal model using COMSOL software is used to compute the in-depth temperature response of the TPS in the presence

of simulated Earth entry external pressure and aerodynamic heating loads.[21,22] Heat transfer is modeled within the TPS layers to predict the time-dependent thermal response given transient surface pressure and heat flux boundary conditions. For the porous layers of the outer fabric and insulative material, heat transfer modes considered include solid conduction, radiation, gas conduction, and advection. The amount of heat and mass that is transferred through the F-TPS layers is calculated from the solution of the local energy and gas mass conservation equations, which are obtained from the theory of gas flow through a porous solid. Further details on the model description can be found in the work by Tobin and Dec.[23]

The TPS is sized to survive heating environments during atmospheric entry while meeting a 400^oC bondline temperature limit with an acceptable level of risk. For reliability analysis, the limit state function can be written as

$$g(\mathbf{x}) = 400 - f(\mathbf{x}) \tag{33}$$

where the temperature limit in the first term is compared to the function evaluation of the thermal model $f(\mathbf{x})$ with uncertain variable vector $\mathbf{x}$. For this problem, the functional form of the limit state is not explicitly known, and as will be shown in this section, can be approximated with SVM. A total of 14 uncertain variables are considered in Table 3 and statistically varied according to the distributions presented by Tobin and Dec.[23] The uncertain variables include thermophysical properties, decomposition, and thickness of insulator layers, and the transient heat flux profile. Two cases will be demonstrated with and without the bondline temperature bias uncertainty due to model inaccuracies compared to test data.

In both cases, a MCS of the COMSOL model with 1999 random samples was performed for a comparison to the SVM analysis. Given that the number of samples for the MCS was small, it is difficult to judge if the simulation converged.[24] Therefore, the failure probability for the MCS is calculated using a 95% confidence interval. The calculation becomes

$$\mathrm{CI}_{\mathrm{MCS}} = \left[p_f - 1.96\sqrt{\frac{1-p_f}{1999}}, p_f + 1.96\sqrt{\frac{1-p_f}{1999}} \right] \tag{34}$$

where p_f is the actual failure probability from the MCS. The above confidence interval holds assuming that the MCS result is used to approximate the average probability of failure. The results for the two cases using the adaptive sampling strategy are presented in Sections A and B.

Table 3: Uncertain variable scale factors varied in the TPS reliability analysis (SF = scale factor)

Parameter	Distribution Type	Mean Value	Standard Deviation or Range
Insulator Virgin Conductivity SF	Normal	1	0.106
Insulator Char Conductivity SF	Normal	1	0.131
Insulator Specific Heat SF	Normal	1	0.0738
Insulator Latent Heat of Reaction	Uniform	1	[0,2]
Cold-wall Heat Flux SF	Normal	1	0.07
Bondline Temperature Bias	Uniform	25	[0,50]
Insulator Layer 1 Thickness SF	Normal	1	0.084
Insulator Layer 1 Density SF	Normal	1	0.083
Insulator Layer 2 Thickness SF	Normal	1	0.084
Insulator Layer 2 Density SF	Normal	1	0.083
Insulator Layer 3 Thickness SF	Normal	1	0.084
Insulator Layer 3 Density SF	Normal	1	0.083
Insulator Layer 4 Thickness SF	Normal	1	0.084
Insulator Layer 4 Density SF	Normal	1	0.083

A. Unbiased Case

For this case, 13 uncertain variables are considered and does not account for the bias uncertainty parameter in the bondline temperature. The initial LCVT DOE consists of 30 samples that ranged out toward 3σ from the mean for each parameter. For the model ϵ_1 and ϵ_2 were chosen to be 0.06 and 0.005, respectively. Other parameters include, the number of test points, which was $N_{\text{test}} = 1,000,000$ and the number of maximum samples per iteration, which was $N_{\text{run}} = 7$. The adaptive sampling routine added 84 points for 114 overall COMSOL evaluations. The MCS failure probability was found to be $p_f = 0.216$. Then, applying Eq. (34), the 95% confidence interval becomes $[0.195, 0.231]$. The results are given in Table 4 and convergence criterion can be visualized in Figure 11.

Table 4: Sample size and failure probabilities for unbiased TPS problem with $\epsilon_1 = 0.06$ and $\epsilon_2 = 0.005$.

N_{initial}	N_{added}	COMSOL	Adaptive SVM-based	
		MCS ($N = 1999$)	MCS	FORM
30	84	[0.195,0.231]	[0.19,0.21]	0.210

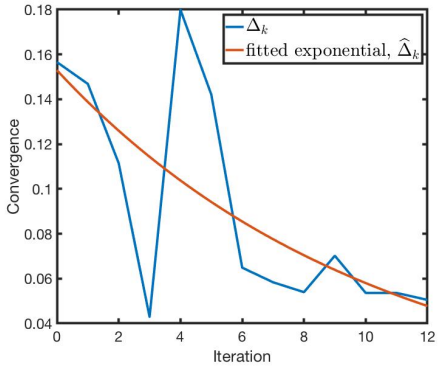


Figure 11: Unbiased case: number iterations vs. convergence criterion

The failure probability for the adaptive sampling SVM-based MCS is bounded between [0.19, 0.21]. The probability is bounded in this way due to a ringing effect in the failure rate when the program was close to convergence. The probability bounced between 0.19 and 0.21 during iterations 11 through 14, as shown in Figure 12. The adaptive sampling probability failure lies in the interval $[0.19, 0.21]$, which is nearly within the 95% confidence bound of $[0.195, 0.231]$ from the MCS of the COMSOL model. Here, there is no COMSOL based model for FORM, because the limit state is unknown, therefore FORM would have been difficult to converge. However, the adaptive SVM-based FORM probability of failure lies within the 95% confidence interval. This illustrates the adaptive sampling strategy can perform an accurate estimate for the TPS reliability while requiring far fewer COMSOL evaluations when using a direct MCS.

B. Biased Case

For this case, 14 uncertain variables are considered, including the bias uncertainty parameter that ranges $[0, 50]$ to the bondline temperature. The initial LCVT DOE consists of 30 samples that ranged out toward 3σ from the mean for each parameter. For the model ϵ_1 and ϵ_2 were chosen to be 0.05 and 0.004, respectively. The initial sample size is 30 and there are 80 samples added through the routine. This gives a total of 110 COMSOL evaluations. Other parameters include, the number of testing points, $N_{\text{test}} = 1,000,000$ and number of maximum samples per iteration, $N_{\text{run}} = 7$. A MCS of COMSOL gives a failure probability of $p_f = 0.393$. Then, applying Eq. (34), the 95% confidence interval was determined to be $[0.383, 0.427]$. The results are given in Table 4 and convergence criterion can be visualized in Figure 13.

The probability of failure using a MCS on the SVM is bounded between [0.38, 0.41], which is due to a similar ringing effect as in the unbiased case. The bouncing between 0.38 and 0.41 happens around iterations

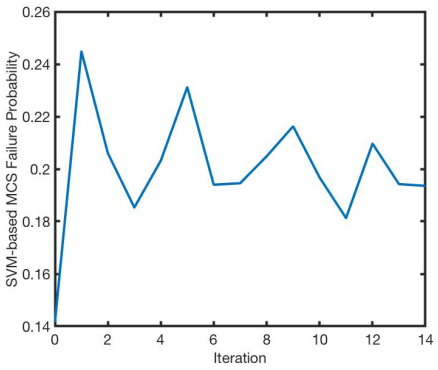


Figure 12: Unbiased case: number iterations vs. adaptive SVM-based MCS failure probability

Table 5: Sample size and failure probabilities for unbiased TPS problem with $\epsilon_1 = 0.05$ and $\epsilon_2 = 0.004$.

N_{initial}	N_{added}	COMSOL	Adaptive SVM-based	
		MCS ($N = 1999$)	MCS	FORM
30	80	[0.383,0.427]	[0.38,0.41]	0.423

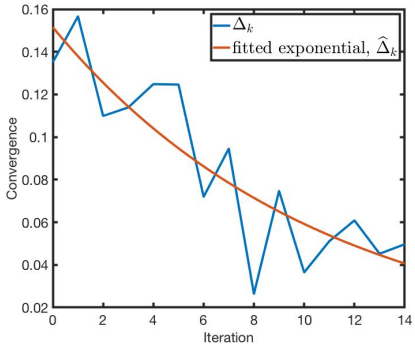


Figure 13: Biased case: number iterations vs. convergence criterion

12 through 16, which can be seen in Figure 14. Given the adaptive sampling failure probability lies between [0.39, 0.41], it lies nearly between the 95% confidence bound of [0.383,0.427] for the actual probability of failure. The adaptive SVM-based FORM failure probability also lies within the 95% confidence interval. Hence, the SVM generated here gives an accurate approximation of the limit state and, in return, the reliability of the TPS while requiring far fewer COMSOL evaluations using a direct MCS.

IV. Conclusion

The objective of this work was to formulate a computationally efficient approach to reliability analysis using support vector machines coupled with an adaptive sampling routine to minimize the number of model evaluations.The adaptive sampling strategy discussed here displays an efficient way to add samples around the limit state in order to accurately predict the failure probability. As opposed to Monte Carlo simulations, this methodology only needs a small number of function evaluations to give an approximate failure probability. The proposed strategy was applied to known limit state and unknown limit state problems, including the reliability analysis of a thermal protection system. In each case, the predicted failure probability aligned with the actual failure probability giving indication that the proposed methodology is well suited for engineering design and analysis of thermal protection systems.

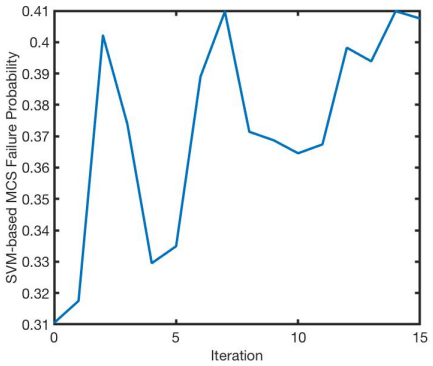


Figure 14: Biased case: number iterations vs. adaptive SVM-based MCS failure probability

For future work, the stopping criterion needs to be investigated rigorously. If both ϵ_1 and ϵ_2 are too small, then it could lead to adding more data than necessary. If ϵ_1 and ϵ_2 are too large, it could lead to inaccuracies in the prediction of failure probability. However, given sufficient stopping criterion the SVM can reliability predict a failure probability. The stopping criterion is model dependent and research on how to pick an reliable criterion based upon the model remains for further investigation. For other future work, more analysis remains on the SVM kernel. For nonlinear data the accuracy of the SVM is completely dependent on the kernel. Given that there is no general knowledge of the behavior of the limit state for most problems it is important to formulate an accurate way to pick the kernel. Research into cross-validation models for varying kernels will be applied to this problem in forthcoming works.

References

[1]Rackwitz, R., "Reliability analysisa review and some perspectives," *Structural safety*, Vol. 23, No. 4, 2001, pp. 365–395.

[2]Li, H.-s., Lü, Z.-z., and Yue, Z.-f., "Support vector machine for structural reliability analysis," *Applied Mathematics and Mechanics*, Vol. 27, 2006, pp. 1295–1303.

[3]Basudhar, A. and Missoum, S., "Adaptive explicit decision functions for probabilistic design and optimization using support vector machines," *Computers & Structures*, Vol. 86, No. 19, 2008, pp. 1904–1917.

[4]Basudhar, A., Missoum, S., and Sanchez, A. H., "Limit state function identification using support vector machines for discontinuous responses and disjoint failure domains," *Probabilistic Engineering Mechanics*, Vol. 23, No. 1, 2008, pp. 1–11.

[5]Song, H., Choi, K. K., Lee, I., Zhao, L., and Lamb, D., "Adaptive virtual support vector machine for reliability analysis of high-dimensional problems," *Structural and Multidisciplinary Optimization*, Vol. 47, No. 4, 2013, pp. 479–491.

[6]Mahadevan, S., "Physics-based reliability models," *MECHANICAL ENGINEERING-NEW YORK AND BASEL-MARCEL DEKKER-*, 1997, pp. 197–232.

[7]Shinozuka, M., "Basic analysis of structural safety," *Journal of Structural Engineering*, Vol. 109, No. 3, 1983, pp. 721–740.

[8]Wang, Y., Yu, X., and Du, X., "Improved Reliability-Based Optimization with Support Vector Machines and Its Application in Aircraft Wing Design," *Mathematical Problems in Engineering*, Vol. 2015, 2015.

[9]Zhao, Y.-G. and Ono, T., "A general procedure for first/second-order reliabilitymethod (form/sorm)," *Structural safety*, Vol. 21, No. 2, 1999, pp. 95–112.

[10]Rosenblatt, M., "Remarks on a multivariate transformation," *The annals of mathematical statistics*, Vol. 23, No. 3, 1952, pp. 470–472.

[11]Vapnik, V., *The nature of statistical learning theory*, Springer Science & Business Media, 2013.

[12]Vapnik, V. N., "An overview of statistical learning theory," *IEEE transactions on neural networks*, Vol. 10, No. 5, 1999, pp. 988–999.

[13]Cortes, C. and Vapnik, V., "Support-vector networks," *Machine learning*, Vol. 20, No. 3, 1995, pp. 273–297.

[14]Cristianini, N. and Shawe-Taylor, J., *An introduction to support vector machines and other kernel-based learning methods*, Cambridge university press, 2000.

[15]Romero, V. J., Burkardt, J. V., Gunzburger, M. D., and Peterson, J. S., "Comparison of pure and Latinized centroidal Voronoi tessellation against various other statistical sampling methods," *Reliability Engineering & System Safety*, Vol. 91, No. 10, 2006, pp. 1266–1280.

[16]Kohavi, R. et al., "A study of cross-validation and bootstrap for accuracy estimation and model selection," *Ijcai*, Vol. 14, Montreal, Canada, 1995, pp. 1137–1145.

[17]"MATLAB Statistics and Machine Learning Toolbox," 2017.

[18]Friedman, J. H., Bentley, J. L., and Finkel, R. A., "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 3, No. 3, 1977, pp. 209–226.

[19]Guan, X. and Melchers, R., "Effect of response surface parameter variation on structural reliability estimates," *Structural Safety*, Vol. 23, No. 4, 2001, pp. 429–444.

[20]Hurtado, J. E. and Alvarez, D. A., "Classification approach for reliability analysis with stochastic finite-element modeling," *Journal of Structural Engineering*, Vol. 129, No. 8, 2003, pp. 1141–1149.

[21]COMSOL, "COMSOL Multiphysics User's Guide, v5.0," Tech. rep., COMSOL/CM-010004, Oct. 2014.

[22]Sullivan, R. M., Baker, E. H., and Dec, J. A., "Analysis Methods for Heat Transfer in Flexible Thermal Protection Systems for Hypersonic Inflatable Aerodynamic Decelerators," Tech. rep., NASA/TM-2014-218311, April 2014.

[23]Tobin, S. A. and Dec, J. A., "A Probabilistic Sizing Demonstration of a Flexible Thermal Protection System for a Hypersonic Inflatable Aerodynamic Decelerator, AIAA 2015-1895," *AIAA SciTech Conference*, Kissimmee, FL, Jan. 2015.

[24]Robert, C. P., "M onte C arlo Methods," *Wiley StatsRef: Statistics Reference Online*, 2014, pp. 1–13.