Journal of Machine Learning Research 24 (2023) 1-30 Submitted 9/21; Revised 11/22; Published 1/23

A Randomized Subspace-based Approach for Dimensionality Reduction and Important Variable Selection

Di Bo DBO@VOLS.UTK.EDU
Hoon Hwangbo HHWANGB1@UTK.EDU
Industrial & Systems Engineering
The University of Tennessee, Knoxville
Knoxville, TN 37996-2315, USA

Vinit Sharma VINIT.SHARMA@UTK.EDU
National Institute for Computational Sciences
The University of Tennessee, Knoxville
Oak Ridge, TN 37831-6173, USA

Corey Arndt CARNDT@VOLS.UTK.EDU
Stephanie TerMaath STERMAAT@UTK.EDU
Mechanical, Aerospace, and Biomedical Engineering
The University of Tennessee, Knoxville
Knoxville, TN 37996-2010, USA

Editor: Isabelle Guyon

Abstract

An analysis of high-dimensional data can offer a detailed description of a system but is often challenged by the curse of dimensionality. General dimensionality reduction techniques can alleviate such difficulty by extracting a few important features, but they are limited due to the lack of interpretability and connectivity to actual decision making associated with each physical variable. Variable selection techniques, as an alternative, can maintain the interpretability, but they often involve a greedy search that is susceptible to failure in capturing important interactions or a metaheuristic search that requires extensive computations. This research proposes a novel method that identifies critical subspaces, reduced-dimensional physical spaces, to achieve dimensionality reduction and variable selection. We apply a randomized search for subspace exploration and leverage ensemble techniques to enhance model performance. When applied to high-dimensional data collected from the failure prediction of a composite/metal hybrid structure exhibiting complex progressive damage failure under loading, the proposed method outperforms the existing and potential alternatives in prediction and important variable selection.

Keywords: Subspace-based modeling, randomized algorithms, feature selection, hybrid material analysis, damage tolerance modeling

1. Introduction

With the recent breakthrough in computing, a real-world data analysis tends to involve voluminous high-dimensional data sets for modeling various systems with great complexity (Reddy et al., 2020; Zhu et al., 2020). An increased dimensionality can provide more information about an underlying system, but analyzing a high-dimensional data set is not

an easy task due to the curse of dimensionality (Liu, 2019; Zhu et al., 2021). That is, for a fixed sample size, data tend to be sparser in space, so signals are much weaker but noises have far greater impacts, leading to improper analysis outcomes. A common solution to this problem is to evaluate a data-driven model in a meaningful low-dimensional space by identifying a subset of features that can represent the entire data with a minimal loss of information, e.g., through dimensionality reduction or variable selection (Liu, 2019).

Dimensionality reduction extracts inherent features from high-dimensional data and relocates the data in a low-dimensional space constructed by the extracted features, alleviating the curse of dimensionality (Van Der Maaten et al., 2009). However, existing dimensionality reduction techniques suffers from the lack of interpretability as the extracted features are "artificial" having obscure connections to the original "physical" variables (Van Der Maaten et al., 2009). In engineering applications, identifying important physical variables is crucial for reducing the number of experimental runs required to optimize a system of interest. In this context, the usage of dimensionality reduction is limited. As an alternative, variable selection, also referred to as subset selection or feature selection, can distinguish important physical variables by selecting a subset of the original variables that are significant (Wei et al., 2015b). Finding an exact optimal subset is computationally intractable, so variable selection typically involves a heuristic search (Fong et al., 2013). A simple heuristic such as a greedy approach cannot ensure the quality of the final model, and a complex heuristic such as genetic algorithm is computationally expensive. In addition, while applying a heuristic search, important interactions between physical variables can easily be missed.

This paper develops a randomized subspace-based modeling for the purpose of alleviating the curse of dimensionality and providing valuable insight about an underlying system for system optimization. To this end, we model a regression problem as an ensemble of multiple base learners where each base learner is defined over a lower-dimensional input space only; we will refer to this reduced-dimensional physical space as a subspace. The subspace-based modeling is similar to general variable selection approaches as it selects subsets of the original variables to achieve dimensionality reduction. However, it differs from them as it leverages multiple distinct subsets and accordingly multiple models to evaluate a function, extending the idea of stepwise regression (Johnsson, 1992) and stacking (Yao et al., 2018). To balance between model accuracy and computational complexity, we randomly generate subspaces and selectively choose important subspaces. By construction, an important subspace informs that not only the variables therein but also the potential interactions between them are significant. We prescribe the subspaces to be a fixed (low) dimension considering that a high-order interaction exceeding a certain order is rarely significant in practice. By keeping all subspaces at a low dimension, functional evaluation is always performed at a low-dimensional space without a risk of suffering from the curse of dimensionality.

The potential of this method to significantly impact our characterization and understanding of advanced engineering systems is demonstrated using the challenging problem of identifying the most influential material properties on damage tolerance for a layered hybrid structure and formulating a reduced order model based on these most sensitive parameters. This example was chosen due to the high dimensional parameter space and wide range of parameter values. Additionally, the high fidelity model used to predict damage tolerance requires a prohibitive amount of computational time to characterize just a small number of parameters in a narrow subspace of the parameter value ranges. The approach presented

herein enables a novel method to rapidly reduce and characterize this vast and complex parameter space to solve a challenging physics-based structural mechanics problem.

The remainder of this article is organized as follows. Section 2 reviews relevant studies on dimensionality reduction and important variable selection. Section 3 presents the proposed subspace-based method by describing model structure, random generation and selective extraction of subspaces, and overall learning process. Section 4 illustrates feature selection and prediction capability of the subspace-based modeling by using popular toy problems and a publicly available data set. Section 5 demonstrates the benefits of the proposed method in comparison with others for modeling the damage tolerance of a hybrid material and discusses its potential usages for structural design, analysis, and optimization. Section 6 concludes the paper and discusses future work.

2. Literature Review

In the past decades, dimensionality reduction has been common in many applications involving a large number of variables, including digital photographs, speech recognition, financial marketing and bioinformatics (Van Der Maaten et al., 2009; Zhu, 2020; Wei et al., 2015a). Dimensionality reduction techniques transform high-dimensional data into a meaningful reduced-dimensional representation, ideally close to its intrinsic dimensions (Van Der Maaten et al., 2009). The intrinsic dimensions of data are the minimum features needed to account for the observed properties of the data (Fukunaga, 2013).

Traditional dimensionality reduction techniques include principal component analysis (PCA), independent component analysis (ICA), and multidimensional scaling (MDS). PCA is one of the most popular linear reduction techniques and dates back to Karl Pearson in 1901 (Pearson, 1901). This technique tries to find orthogonal directions that account for as much variance of data as possible. Due to its attractive advantages of minimal information loss and generation of uncorrelated dimensions, PCA is still popular for use in a broad range of application areas (Zou et al., 2006). For example, Li et al. (2016) applied PCA to evaluate energy security of multiple East Asian countries with respect to vulnerability, efficiency, and sustainability. Salo et al. (2019) proposed a network anomaly detection method based on a reduced dimensionality achieved by combining information gain and PCA. On the other hand, ICA tries to extract independent pieces of information from high-dimensional data, mainly for the purpose of source blind separation that has been popular in signal processing (Hastie et al., 2009). ICA can be useful in other areas of study; for example, Sompairac et al. (2019) discussed the benefits of ICA to unravel the complexity of cancer biology, particularly in analyzing different types of omics data sets. Different from PCA and ICA, MDS is a nonlinear dimensionality reduction technique. It provides a useful graphical representation of data based on the similarity information of individual data points. MDS is a common technique for analyzing network-structured data as presented in Saeed et al. (2019) that applied MDS to wireless networks localization.

Over the recent decades, many nonlinear or nonparametric dimensionality reduction techniques have been proposed (Lee and Verleysen, 2007; Saul et al., 2006). Kernel PCA is a reformulation of traditional linear PCA constructing feature space through a kernel function (Schölkopf et al., 1998). Choi et al. (2005) found PCA was inefficient and problematic for modeling a nonlinear system but kernel PCA effectively captured nonlinearity

while achieving dimensionality reduction. Xu et al. (2019) proposed a defect prediction framework that combined kernel PCA and weighted extreme learning machine to extract representative data features and learn an effective defect prediction model. Tenenbaum et al. (2000) proposed an isometric feature mapping (Isomap) technique that was based on classical MDS but sought to retain the intrinsic geometry of data sets as captured in the geodesic manifold distances. It achieves the goal of nonlinear dimensionality reduction by using easily measured local metric information to learn the underlying global geometry of a data set. Hinton and Salakhutdinov (2006) suggested using deep autoencoder networks while transforming high-dimensional data into low-dimensional codes by training a multilayer neural network with a small central layer. They presented that the deep autoencoder networks outperformed PCA with a proper selection of initial weights. Belkina et al. (2019) introduced t-distributed stochastic neighbor embedding (t-SNE), and Gisbrecht et al. (2015) presented kernel t-SNE as an extension of t-SNE to a parametric framework, which enabled explicit out-of-sample extensions. They demonstrated that kernel t-SNE yielded satisfactory results for large data sets. McInnes et al. (2018) proposed uniform manifold approximation and projection (UMAP) constructed by a theoretical framework based on Riemannian geometry and algebraic topology. Compared to t-SNE, UMAP arguably preserves more of the global structure. Since UMAP does not have computational restrictions on embedding dimension, it can be easily used for general dimensionality reduction.

Even with the theoretical and methodological advance of the dimensionality reduction techniques, they cannot pinpoint which variables are important among the existing variables; instead, they extract inherent features that are artificial. Determining important physical variables is a critical task in many experimental studies, including those for structural mechanics (TerMaath, 2018), environmental science, and bioinformatics (Wei et al., 2015b). This is because the knowledge of important variables can suggest which variables to test and optimize for subsequent experiments to reduce the number of experimental trials and hence expedite the overall experimental procedure. In this context, variable selection (or feature selection) that extracts a subset of existing variables can be a good alternative to general dimensionality reduction since it is capable of maintaining the original variable structure while reducing the dimensionality.

Feature selection methods are typically classified into three groups of filter methods, wrapper methods, and embedded methods (Chandrashekar and Sahin, 2014; Dhyaram and Vishnuvardhan, 2018). Filter methods evaluate each variable based on some ranking criteria, such as Pearson correlation coefficient (Battiti, 1994) or mutual information (Torkkola, 2003). Wrapper methods, evaluating different subsets of variables by using a learning algorithm, generally provide better performance compared to filter methods (Xue et al., 2015). To explore potential subsets of variables, an exhaustive search requires considering 2^p different feature combinations when there are p variables, which is impractical. Instead, sequential forward selection (Whitney, 1971) and sequential backward selection (Marill and Green, 1963) algorithms that apply greedy search have been used broadly. More recently, sequential floating forward selection (Pudil et al., 1994) and adaptive sequential floating forward selection (Somol et al., 1999) have been proposed to alleviate poor performance of the greedy search. However, these sequential selection methods still suffer from a nesting problem caused by the nature of the greedy search, so the selection outcome is generally far from the optimum. To overcome the nesting problem, heuristic search methods, e.g., one

based on genetic algorithm (Alexandridis et al., 2005), have been proposed. These wrapper methods, however, require an extensive amount of computation and are prone to overfitting (Chandrashekar and Sahin, 2014). Embedded methods aim to alleviate the computational cost of wrapper methods by integrating feature selection and model learning into a single process (Xue et al., 2015). In other words, some ranking criteria, including max-relevancy min-redundancy (Peng et al., 2005) and the weights of a classifier (Mundra and Rajapakse, 2009), are used for the initial feature reduction, and then wrapper methods are applied for the final feature selection.

These days, machine learning methods, such as random forest (RF) (Kursa, 2014; Chen et al., 2020; Zhou and Hooker, 2021) and neural network (NN) (Olden et al., 2004) have been used to measure variable importance and rank variables based on model's prediction accuracy, which can be easily extended for feature selection (Louppe et al., 2013; Degenhardt et al., 2019). Gregorutti et al. (2017) explored the usage of RF in the presence of correlated predictors. Their results motivated the use of the recursive feature elimination algorithm that uses the permutation importance measure as a ranking criterion. Chen et al. (2020) compared results from three popular data sets (bank marketing, car evaluation database, human activity recognition using smartphones) with and without feature selection based on multiple methods, including RF and recursive feature elimination. The experimental results demonstrated that RF achieved the best performance in all experimental groups. Zhou and Hooker (2021) considered split-improvement, a popular feature importance measure for tree-based models, for an RF model to determine variable importance. As split-improvement suffers from bias towards continuous features, they resolved this issue and demonstrated the effectiveness of their solution approach both theoretically and empirically. On the other hand, Olden et al. (2004) compared nine variants of artificial neural network (ANN) for quantifying variable importance from simulated data. For this comparison, the true importance of variables was known and used for verification. Their results showed the connection weight approach provided the best accuracy and it was the only method that correctly identified the rank of variable importance. Liu (2019) presented a deep Bayesian rectified linear unit (ReLU) network for high-dimensional regression and variable selection. The result showed their method outperformed existing classic and other NN-based variable selection methods.

Although these variable selection methods provide an efficient tool to extract important variables, they primarily focus on the importance of individual variables, with little consideration for the importance of feature interactions. In a general experimental problem, however, knowledge about the significance of feature interactions is a critical factor for designing experiments. Instead of merely finding important individual variables, our approach aims to identify critical subspaces, each of which presents which variable combination as a whole (including interactions) is important. For the exploration of potentially important subspaces, a randomized search is employed to improve model accuracy and computational efficiency relative to a greedy search and a metaheuristic search, respectively. Furthermore, our method leverages multiple subsets of variables (subspaces) for model construction instead of relying on a single subset as the existing methods do, which can provide a more flexible model. The details of the proposed method will be discussed in the subsequent sections.

3. Subspace-Based Modeling and Critical Subspaces

In this section, we develop subspace-based modeling for solving a general supervised learning problem (in particular, a regression problem) and identifying critical variables and interactions. The data set of interest contains data pairs of $\{\mathbf{x}_i, y_i\}_{i=1}^n$ where $\mathbf{x}_i$ is a p-dimensional input vector and y_i is the corresponding response. For subspace-based modeling, we form a subspace, a space spanned by a subvector of input $\mathbf{x}$, determine the significance of a subspace for modeling response, and use such a critical subspace as a basic unit for model building. A critical subspace implies that the variables forming the space and their interactions are important altogether, so it naturally pinpoints important variables and interactions. In what follows, we describe the proposed model structure, a randomized search for subspace generation, a significance evaluation of a subspace, and the overall model learning process.

3.1 Subspace-based Model

Considering a multi-inputs, single output regression problem, we estimate a function $f : \mathbb{R}^p \to \mathbb{R}$ that relates p-dimensional input $\mathbf{x}$ and output y as $y = f(\mathbf{x}) + \varepsilon$ where ε is an additive noise. We model the unknown function f as an additive mixture of subfunctions $g_j : \mathbb{R}^k \to \mathbb{R}$ for $j = 1, \ldots, J$ defined in a lower dimensional space of dimension $k \ll p$:

$$y = f(\mathbf{x}) + \varepsilon \approx g_1(\mathbf{z}_1) + g_2(\mathbf{z}_2) + ... + g_J(\mathbf{z}_J) + \varepsilon, \tag{1}$$

where $\mathbf{z}_j$ is the jth subvector of $\mathbf{x}$ of size k. Each $\mathbf{z}_j$ for $j = 1, \ldots, J$ takes different components of $\mathbf{x}$. For example, suppose $k = 3$ and $\mathbf{x} = \left[x_1, x_2, \ldots, x_p\right]^T$ where $p > 20$. Then, we may have $\mathbf{z}_1 = \left[x_3, x_8, x_{12}\right]^T$ and $\mathbf{z}_2 = \left[x_7, x_{11}, x_{20}\right]^T$. We define a reduced-dimensional space spanned by each subvector $\mathbf{z}_j$ as a subspace, and g_j estimates the response y within a subspace formed by $\mathbf{z}_j$. As such, in Eq. (1), we model the function f as a mixture of subspace-based models. This ensemble extends the idea of stepwise regression (Johnsson, 1992) and stacking (Yao et al., 2018) as different subsets of the features are used to build distinct models (g_j) predicting the response.

The advantage of the subspace-based modeling is obvious. By evaluating y in low-dimensional subspaces, there is no risk of suffering from the curse of dimensionality, which is not true when evaluating y in a single high-dimensional space. Different from general dimensionality reduction methods, the subspace-based modeling does not require extracting artificial dimensions, but the reduction of dimensionality is achieved by forming low-dimensional physical spaces. One major shortcoming of the subspace-based modeling is its incapability of modeling interactions of an order higher than k. However, in many real-world problems, a high-order interaction is almost negligible in modeling response. From a preliminary study, we found that $k = 3$, i.e., modeling up to 3-factor interactions, produced decent predictions; see Appendix B for more details.

The key to the success in the subspace-based modeling lies in how to form the subspaces, i.e., how to generate the subvectors of $\mathbf{z}_j$ for $j = 1, \ldots, J$. For an exhaustive search, if we assume $p = 41$ and $k = 3$, the number of all subspace candidates is $\binom{41}{3} = 10660$. Evaluating models for this many subspaces and determining whether to include each of them requires a considerable amount of computation (still, much smaller compared to 2^{41} for an exhaustive search in the absence of the proposed model structure in Eq. (1)). This requires an efficient

strategy for subspace exploration, and in the next section, we discuss how to generate subspaces and how to determine critical subspaces.

3.2 Subspace Generation and Extraction

To explore a broad area covering potential subspace configurations, we generate subspaces randomly. In particular, at each draw, we randomly choose k variables out of p available and evaluate whether to include this randomly generated subspace into the model or not. The sampling process is without replacement, but we allow duplicated selection of a single variable at multiple draws. In other words, an input variable x_1 could be a part of subspace $\mathbf{z}_1$, and the variable can be chosen again at later draws for $\mathbf{z}_j$ for $j > 1$. This is to ensure that multiple interactions associated with a key variable are not lost.

We determine the significance of a randomly generated subspace by evaluating the percentage reduction in prediction error measured by root-mean-square error (RMSE). To evaluate out-of-sample error and avoid possible overfitting, we apply 5-fold cross validation (CV) for error estimation. This 5-fold CV is applied to the data set assigned for model learning (excluding testing portion), and the data set is split into 5 small subsets of the data. Each subset serves as a validation data set whereas the remaining four subsets are collectively used to train a model. In this way, we generate 5 different train/validation data sets, as shown in Fig. 1.

To test the significance, a temporary subspace-based model (g_t) for a randomly generated subspace ($\mathbf{z}_t$) will be added into the model, and the prediction error of this extended model will be evaluated in terms of the out-of-sample RMSE. The temporary subspace-based model, g_t where t denotes the current iteration of the subspace generation, is learned for each of the five different train data sets, and the prediction error of the extended model is

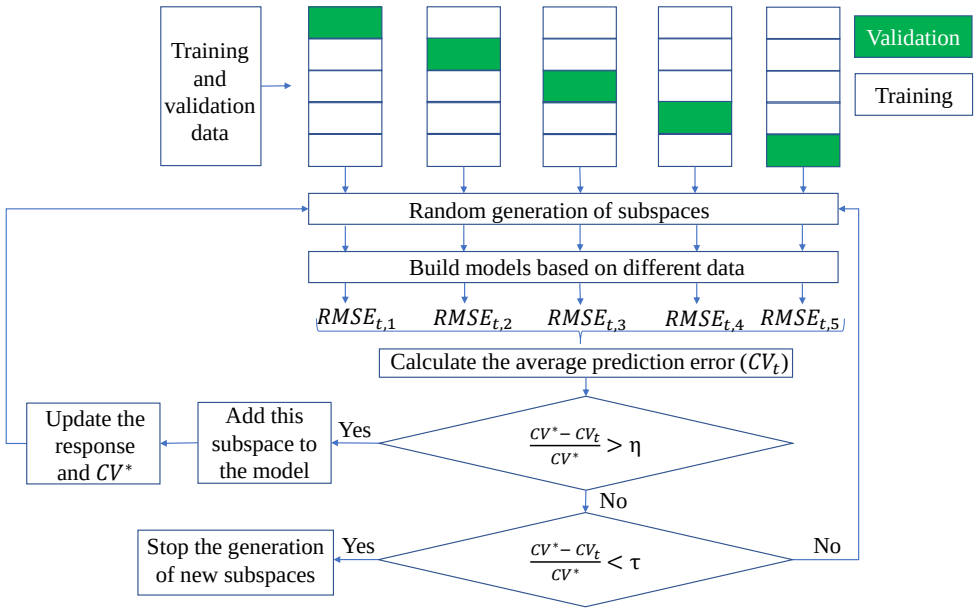

Figure 1: A flow chart describing the process of subspace generation and extraction

calculated by using the corresponding five validation data sets. This will produce $RMSE_{t,q}$ for $q = 1, \ldots, 5$ for each train/validation split at iteration t. To be specific, the RMSE is calculated as

$$RMSE_{t,q} = \sqrt{\frac{1}{n_{v_q}} \sum_{\{\mathbf{x}_i, y_i\} \in \mathcal{V}_q} (y_i - \tilde{f}_t(\mathbf{x}_i))^2}, \tag{2}$$

where $\mathcal{V}_q$ is the validation data set for the qth train/validation split and n_{v_q} is the number of observations in the validation data set. $\tilde{f}_t(\mathbf{x}_i) = \sum_{j \in \mathcal{J}^*} g_j(\mathbf{z}_j) + g_t(\mathbf{z}_t)$ where $\mathcal{J}^*$ is the set of indices for the selected subspaces by the previous iteration, $t-1$. The CV score is then the average of the five prediction errors, i.e., $CV_t = \sum_{q=1}^{5} RMSE_{t,q}/5$.

We use the percentage reduction of the prediction error for the selection of a critical subspace and also for the termination of the iterative search for new subspaces. The percentage error reduction is defined as $\Delta e = (CV^* - CV_t)/CV^*$ where CV^* is the minimum (best) CV score obtained until the previous iteration. The initial value of CV^* is calculated from a model that only includes a constant term (the average of the five sample means of y_i's in each training data set). We consider a subspace significant if $\Delta e > \eta$ where η is a preset selection threshold. This means that a subspace that reduces the prediction error at least by a certain percentage will be included in the model; otherwise, this subspace is disregarded. The search for a new subspace continues until the percentage reduction Δe becomes less than a termination threshold, τ. If none of the selection criterion and the termination criterion are met, the iterative search continues for the next possible critical subspace. Fig. 1 illustrates the overall process of the subspace generation and extraction.

3.3 Subspace Model Learning and Hyperparameter Selection

This section describes how to estimate each subspace-based model g_t. For this estimation, we use a randomly generated subvector, $\mathbf{z}_t$ as predictors and the current residual calculated before the iteration t as a response to estimate, i.e., using $\tilde{y}_{t,i} = y_i - \sum_{j \in \mathcal{J}^*} g_j(\mathbf{z}_{j,i})$ for $i = 1, \ldots, n$ for response. In other words, we consider a regression problem written as

$$\tilde{y}_{t,i} = g_t(\mathbf{z}_{t,i}) + \tilde{\varepsilon}_{t,i}, \tag{3}$$

where $\mathbf{z}_{t,i}$ is a subvector of $\mathbf{x}_i$, and $\tilde{\varepsilon}_{t,i}$ is a modified noise for $i = 1, \ldots, n$.

To learn the function g_t, we use support vector machine (SVM) as a base learner. For a regression analysis, it is also known as support vector regression (SVR). To train a model based on SVR, we use "ϵ-insensitive" loss function characterized by a flexible tube of ϵ radius. Under this loss function, penalty is assigned only to the points outside the tube and is proportional to the distance from the tube, but no penalty is applied to the points within the tube (Awad and Khanna, 2015); see Eq. (5).

For SVR, the function g_t can be expressed as a linear combination of basis functions, h_m for $m = 1, \ldots, M$, as

$$g_t(\mathbf{z}_t) = \sum_{m=1}^{M} \beta_m h_m(\mathbf{z}_t) + \beta_0, \tag{4}$$

where β_m for $m = 1, \ldots, M$ is a coefficient of each basis function and β_0 is a constant. To estimate g_t, we minimize

$$H(\boldsymbol{\beta}, \beta_0) = \sum_{i=1}^{n} V_\epsilon(\tilde{y}_{t,i} - \hat{g}_t(\mathbf{z}_{t,i})) + \frac{\lambda}{2} \sum_{m=1}^{M} \beta_m^2 \\ \text{where } V_\epsilon(r) = \begin{cases} 0, & \text{if } |r| < \epsilon, \\ |r| - \epsilon, & \text{otherwise,} \end{cases} \tag{5}$$

where $\boldsymbol{\beta} = \left[\beta_1, \ldots, \beta_M\right]^T$ is a coefficient vector, and λ is a regularization parameter. The minimizer of Eq. (5) provides the estimate of g_t in the form of

$$\hat{g}_t(\mathbf{z}) = \sum_{i=1}^{n} \alpha_i K_t(\mathbf{z}, \mathbf{z}_{t,i}), \tag{6}$$

where α_i for $i = 1, \ldots, n$ is a Lagrangian dual variable for the Lagrangian primal shown in Eq. (5), and $K_t(\cdot, \cdot)$ is a kernel function that represents the inner product of the unknown basis functions, h_m for $m = 1, \ldots, M$ (Hastie et al., 2009).

To fully specify the estimate of g_t, some hyperparameters need to be determined. This includes the regularization parameter, λ (related to the cost parameter in a typical SVM), the radius of the tube, ϵ, for the loss function, and some kernel related parameters. For this hyperparameter learning, we apply grid search based on generalized cross validation (GCV) criterion. The proposed method already employs 5-fold CV for the selection of critical subspaces and the termination of the iterative search, so another layer of out-of-sample prediction will complicate the data structure, and the training process may suffer from the lack of usable data points. While measuring in-sample error based on training data only, GCV provides a convenient approximation to the leave-one-out CV (Hastie et al., 2009), so it is a proper criterion for the hyperparameter learning of the proposed method. In general, GCV is calculated as

$$GCV(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{y_i - \hat{f}(\mathbf{x}_i)}{1 - \text{trace}(\mathbf{S})/n} \right]^2, \tag{7}$$

where $\mathbf{S}$ is an $n \times n$ hat matrix that performs a linear projection of $\mathbf{y} = \left[y_1, \ldots, y_n\right]^T$ to achieve the estimate of $\mathbf{y}$, i.e., $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$. For the proposed subspace-based model in Eq. (1), deriving this expression of linear projection is not straightforward. Theorem 1 shows a good approximation of $\mathbf{S}$ that can be derived under the assumption of a squared loss function.

Theorem 1. *Assuming a squared loss function, i.e., $V_\epsilon(r) = r^2$, the estimate of the response $\mathbf{y}$ can be expressed as $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y} = (\sum_{j=1}^{J} \mathbf{S}_j)\mathbf{y}$ where $\mathbf{S}_j = \mathbf{S}'_j(\mathbf{I} - \sum_{l=0}^{j-1} \mathbf{S}_l)$, $\mathbf{S}'_j = (\mathbf{K}_j + \lambda\mathbf{I})^{-1}\mathbf{K}_j$, $\mathbf{S}_0 = \mathbf{0}$, $\mathbf{I}$ is an $n \times n$ identity matrix, and $\mathbf{K}_j$ is an $n \times n$ kernel matrix with $\{\mathbf{K}_j\}_{i,i'} = K_j(\mathbf{z}_{j,i}, \mathbf{z}_{j,i'})$ for $i, i' = 1, \ldots, n$.*

Proof See Appendix A. ■

As implied in Theorem 1, we do not allow distinct hyperparameters for each g_j estimation as the GCV calculation is based on the full model. Instead, we assume the hyperparameters to be the same across all g_j for $j = 1, \ldots, J$. We apply a grid search for the hyperparameter

learning, which is to keep hyperparameters at certain levels, add critical subspaces based on this hyperparameter setting, and evaluate the GCV value for the resulting full model (after the termination of subspace search). The set of hyperparameters and the corresponding subspace-based model that minimize the GCV criterion in Eq. (7) will be chosen as an optimal model.

Because the GCV calculation based on the result of Theorem 1 is complicated due to the recursive format, we also consider a simpler version of the trace calculation. For approximation, we replace $\mathbf{S}_j$ by $\mathbf{S}'_j$. The following describes two alternatives we propose for the GCV calculation:

A1. $\text{trace}(\mathbf{S}) = \sum_{j=1}^{J} \text{trace}(\mathbf{S}_j)$ based on Theorem 1.

A2. $\text{trace}(\mathbf{S}) = \sum_{j=1}^{J} \text{trace}(\mathbf{S}'_j)$ for simplification.

The performance of the two alternatives will be compared and discussed in Section 5.

3.4 Overall Algorithm

For clear illustration of the entire learning process, Alg. 1 shows the step-by-step procedures of the proposed subspace-based modeling. To build a model, we use a data set that is dedicated to model learning; if testing is needed (as in Section 5), we split the original data set into two distinct data sets, one for model learning and another for testing. From the model learning data set, we apply 5-fold CV to split training data and validation data where validation data are required to determine the significance of subspaces and the convergence of the algorithm. After the data split, we produce a model that includes a single constant term by using training data only and calculate the prediction error from the validation sets to initialize CV^*. Each hyperparameter setting will be evaluated once a full model is established, i.e., once all critical subspaces are determined and added into the model. For each fixed level set of hyperparameters, we generate a subspace randomly and build an SVR model to estimate g_t. The quality of a subspace is evaluated via a 5-fold CV error of CV_t and its percentage reduction Δe. If a subspace passes the selection criterion, it will be added into the model. This process of generating and evaluating a random subspace will continue until the termination criterion is met. Once the algorithm terminates, the GCV value will be computed according to Eq. (7). After completing all iterations for the grid search, we find the optimal hyperparameter values and apply this optimal setting to the entire model learning data set (training and validation) to build a final model. This final model will be evaluated by a separate test set for comparison with other methods in Section 5. Please note, this entire procedure can be easily adapted to a classification problem by choosing proper loss functions and error measures that are suitable for classification.

4. Comparative Experiments

Prior to applying our method to the hybrid structure data set, in this section, we use three other data sets to evaluate the performance of the proposed method in terms of prediction and feature selection capability. For the first two data sets, the true importance of features and interactions is known, so they are used to assess the feature selection capability. The other data set involves a large-scale feature space where feature selection is essential for

Algorithm 1: Learning process of the randomized subspace modeling

Input: $\mathcal{D}$: data set available for model training and validation; k: predetermined dimension of subspaces; η: selection threshold; τ: termination threshold

Output: Trained model with optimal hyperparameters

Apply 5-fold CV to split given data $\mathcal{D}$ into training ($\mathcal{T}_q$) and validation data ($\mathcal{V}_q$) for $q = 1, \ldots, 5$;

Build a constant model, i.e,. $\hat{y} = \bar{y}$, by using data in $\mathcal{T}_q$ and use this model to predict responses in the corresponding $\mathcal{V}_q$ for $q = 1, \ldots, 5$;

Initialize CV^* as the prediction error of the constant model (the average of 5 RMSEs obtained from $\mathcal{V}_q$ for $q = 1, \ldots, 5$);

Create a grid for assessing different sets of hyperparameters;

for *each level combination of hyperparameters* **do**

  $t \leftarrow 0$; $\mathcal{J}^* \leftarrow \varnothing$; $\mathcal{Z} \leftarrow \varnothing$;

  repeat

    $t \leftarrow t + 1$;

    Randomly draw a k-dimensional subspace, i.e., randomly sample k integers from $\{1, \ldots, p\}$ and store the resulting k-tuple of indices in s;

    By using the randomly drawn subspace, fit a model in Eq. (3) for each $\mathcal{T}_q$ and accordingly, predict $\tilde{y}_{t,i}$ for each $\mathcal{V}_q$ for $q = 1, \ldots, 5$;

    Compute $\tilde{f}_t(\mathbf{x}_i)$ and calculate $RMSE_{t,q}$ and CV_t;

    if $\Delta e > \eta$ **then**

      $CV^* \leftarrow CV_t$;

      $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{t\}$;

      $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{s\}$;

    end

  until $\Delta e < \tau$;

  Use the entire model learning data ($\mathcal{D}$) to build a full model as in Eq. (1) by using the selected set of subspaces, $\mathbf{z}_j$ for $j \in \mathcal{J}^*$ where $\mathbf{z}_j$'s are constructed based on the k-tuples in $\mathcal{Z}$;

  Calculate GCV (either A1 or A2) for the current hyperparameter setting by using the full model;

end

Determine the optimal hyperparameters that minimize GCV;

Return the full model with the minimum GCV (this includes the information of $\mathcal{J}^*$ and $\mathcal{Z}$);

predicting response; we investigate prediction accuracy and robustness through this data set. To demonstrate the effectiveness of the proposed method, we compare its performance with other possible alternatives. The methods subject to comparison include various wrapper methods and machine learning-based methods. Metaheuristic-based methods are not considered here due to their high computational requirement.

4.1 Data Description

The existing studies of feature selection typically concern classification problems and often use the MONK's problem (Dua and Graff, 2019) to assess algorithm performance (Thrun et al., 1991). Three data sets are available for this problem, and for each, different feature combinations are used to generate a response variable. Among the three, we use MONK 1 and MONK 3 data sets but not MONK 2. For MONK 2 data set, all features and 2-factor interactions are designed to be relevant to the response, so the benefit of feature selection achieving dimensionality reduction cannot be clearly shown with this data set. Both MONK 1 and MONK 3 have 6 features and a single response, and they have 556 and 554 data points, respectively. One thing to note is these data sets were designed for a classification problem and the response is a binary variable taking 0 or 1 for two class labels; for the complete data generation process, please refer to Thrun et al. (1991).

Another data set we consider in this section is gene data from Golub et al. (1999). This data set contains 7130 predictors with 72 data records, so it has vast feature space. Instead of exploring over all predictors, we choose 30 predictors based on a relevancy measure to keep the dimensionality manageable but still large enough. Specifically, we calculate the correlation coefficient between each predictor and the response and select 10 predictors from each of three groups of strong, medium, and low relevancy in terms of the linear relationship. This problem is also for classification where the response is a categorical variable with three class labels of 0, 1, and 2.

As we concern a regression problem, we apply variable transformation of the response variable so that the data sets can be more suitable for a regression analysis. In particular, we create a new response variable y from the original categorical response y' as $y = 10y' + v$ where v is a normal random error with mean of 0 and standard deviation of 5. The same treatment is applied to all three data sets we consider, including MONK1, MONK3, and gene data. The scale of 10 and the standard deviation of 5 are chosen as this setting allows non-separable data from the classification perspective and thereby prevents the regression task being too trivial.

4.2 Implementation Detail

By design, the randomized subspace modeling includes a 5-fold CV for the selection of critical subspaces. When comparing prediction performance with other methods, another layer of a 5-fold CV is implemented to have separate test data sets. The similar treatment is applied to other methods being compared. One layer of a 5-fold CV is used to split test data sets, and another layer of a 5-fold CV is used to learn hyperparameters, if applicable. The prediction error is measured by RMSE, and the average and standard deviation of 5 RMSE's are compared. For MONK data sets, they are used to justify the selection of significant features and interactions, there is no need for separate test data, so all data points are used for model learning and feature selection. The other methods subject to comparison include linear regression (LR), lasso regression (Lasso), ridge regression (Ridge), principal component regression (PCR), partial least squares regression (PLS), RF, k-nearest neighbors (k-NN), SVR, and NN to cover various linear and nonlinear modeling commonly used for feature selection. Before applying any method, we normalize the data to prevent any scale-relevant errors.

To implement the randomized subspace modeling, we set the selection and termination thresholds to $\eta = 1\%$ and $\tau = 0.001\%$, respectively, based on the results from preliminary studies; see Appendix B. The dimension of subspaces, k, is set to either 2 or 3 depending on the problem we consider. For the base learner of the subspace-based modeling (SVR), a kernel needs to be chosen among multiple possible options, including linear, polynomial (POLY), radial basis function (RBF), and sigmoid kernels. From preliminary experiments, we found kernels capturing linearity, such as linear and POLY kernels, provided better performance. In this regard, we choose POLY kernel, $K(\mathbf{u}, \mathbf{v}) = (\gamma \mathbf{u}^T \mathbf{v} + \delta)^d$ where γ, δ, and d, respectively, denote the scale, offset, and degree of polynomials, and set $\gamma = 1/k$, $\delta = 0$, and $d = 1$ (found optimal). Since γ had little impact on the final prediction error, we use the default value of an R function for SVR fitting. Considering that we use normalized data set, it is natural to have $\delta = 0$ provide the optimal performance. Although $d = 1$ yields the optimal result, the POLY kernel is chosen over the linear kernel as it is more flexible and capable of modeling nonlinearity with different parameterization. On the other hand, SVR learning itself involves some parameters, ϵ and $C := 1/\lambda$. We test three levels for each parameter, i.e., $\epsilon = 0.01, 0.1, 0.5$ and $C = 1, 2, 5$, and this generates nine distinct parameter settings among which an optimal setting is determined. When applying a grid search for this hyperparameter learning, if there exists a poor level combination, the designed termination criterion may not work properly. To prevent iterating more than what is required for the exhaustive search, we force the search process to stop after a certain number of iterations. Assuming three-dimensional subspaces, for the MONK's data sets involving 6 features, we use 20 ($\binom{6}{3} = 20$) for the maximum number of iterations. Similarly, for the gene data, we use 4060 ($\binom{30}{3} = 4060$) to limit the number of iterations.

Excluding NN and the proposed method, all methods are executed in R by using `train()` function in the `caret` package. Chen et al. (2020) show `varImp()` function (with RF) is an efficient tool for calculating variable importance, so we use the same function to measure variable importance for other methods. NN is implemented in Python by using `keras` package. To make the result comparable to the existing variable selection methods based on NN (Olden et al., 2004; Liu, 2019), we test over multiple NN structures, one or two hidden layer(s), and three activation functions of linear, sigmoid, and ReLU. We also evaluate various hyperparameter options including the numbers of neurons, batch size, and epochs. With all this variation, we pick an NN model with the lowest prediction error for the result comparison. For NN, variable importance is calculated by the Connection Weight Approach, referring to the experimental results in Olden et al. (2004). Since all these methods rely on variable importance scores for feature selection, we list the first twenty variables with high importance score for each 5-fold learning and find the common variables selected shown in the 5 lists to report critical individual variables.

4.3 Experimental Results

From the MONK's problem, we investigate if feature selection methods can accurately identify important features (and interactions) and also compare the running time of feature selection alternatives. Among 6 variables available in the MONK data sets, i.e., a_i for $i = 1, \ldots, 6$, (a_1, a_2) and a_5 are the significant features in MONK 1 where the parentheses signify interactive features. In MONK 3, (a_2, a_5) and (a_4, a_5) are designed to be significant.

Since there are three significant variables in MONK 1, we set the dimension of subspaces to 3, i.e., $k = 3$. MONK 3, on the other hand, includes two interactive features of dimension of 2, so we set $k = 2$ for this data set.

The result of the feature selection and computing time is shown in Table 1. For MONK 1, the critical subspace captured by our method is (a_1, a_2, a_5). The ideal selection is to choose (a_1, a_2) and a_5 separately. The proposed method imposes a fixed dimension on subspaces, and this limits its flexibility in selecting features. Still, the selected three-dimensional subspace involves all target features, including the interaction between a_1 and a_2. Among all alternatives, only the proposed method and RF identify the correct variables for significant features. Other methods miss one or two important variables. Please note, the other methods including the RF-based feature selection are not capable of specifying any significant interaction. The presented results merely show their selection of individual variables. For MONK 3, our method correctly selects the two sets of interactive features, i.e., (a_2, a_5) and (a_4, a_5). In addition, the proposed method is the only method identifying all significant variables while the others miss one important variable, a_4.

As can be expected, linear models do not require a large amount of computation; running each of them can be completed in 3 seconds for this particular analysis. Among nonlinear models, k-NN and SVM with a polynomial kernel achieve comparable computational time marking slightly greater than 3 seconds. The other nonlinear models require more computations in general. Our method's running time is greater than RF's but still comparable in scale; also much less than NN's computational time. Although the proposed method requires more computations, it is worthwhile considering the feature selection results.

The comparison of prediction capability from the gene data is shown in Table 2. For this particular data set, in general, nonlinear models perform slightly better than linear

	MONK 1		MONK 3	
Method	Sig. features	Time (s)	Sig. features	Time (s)
LR	a_5, a_6	2.81	a_2, a_5	2.07
Lasso	a_5	2.36	a_2, a_5	2.37
Ridge	a_5, a_6	2.31	a_2, a_5	2.39
PCR	a_5	2.96	a_2, a_5	2.18
PLS	a_5	2.28	a_2, a_5	2.3
RF	a_1, a_2, a_5	38.38	a_2, a_5	37.57
k-NN	a_5	3.30	a_2, a_5	2.33
SVM (RBF)	a_5	12.63	a_2, a_5	12.55
SVM (POLY)	a_5	4.07	a_2, a_5	4.06
NN	a_2, a_5	720.00	a_2, a_5	840.00
Subspace-SVM	(a_1, a_2, a_5)	**51.50**	(a_2, a_5), (a_4, a_5)	**60.72**
Ground Truth	(a_1, a_2), a_5		(a_2, a_5), (a_4, a_5)	

Table 1: Comparison of the feature selection results and running time for the MONK's Problem

Method	Average	Std. dev.
LR	11.91	1.22
Lasso	9.92	1.54
Ridge	9.59	1.44
PCR	9.55	1.62
PLS	10.28	1.72
RF	9.43	1.75
k-NN	9.41	1.54
SVM (RBF)	9.5	1.38
SVM (POLY)	9.55	1.40
NN	10.87	1.69
Subspace-SVM	**9.41**	**1.21**

Table 2: Comparison of the average and standard deviation of RMSE's from 5-fold CV evaluation (gene data)

counterparts in terms of the average of 5 test errors calculated from the 5-fold CV. Although NN has relatively poor accuracy, this can be attributed to the lack of data records; note, there are only 72 observations in this data set. The standard deviations (calculated from 5 test errors) are similar between linear and nonlinear models. Among all the alternatives, k-NN obtains the lowest average prediction error with relatively high standard deviation while LR achieves the smallest standard deviation with the highest prediction error. On the other hand, the proposed method attains not only the lowest prediction error but also the lowest standard deviation. This shows that the proposed method not only has strength in feature selection but also attains decent predictions.

5. Damage Tolerance Analysis of Hybrid Materials

Design and analysis of layered hybrid structure is challenged by the many possible choices of materials and configurations. This vast parameter space is impractical to explore through physical testing and is computationally prohibitive due to the analysis time required due to the large number and ranges of input parameters. The proposed method is applied to overcome this limitation and provides an efficient and accurate approach to computationally characterize the parameter space and informs limited physical testing to define parameters. The reduced order model can then be used to rapidly explore the parameter space to customize and optimize layered designs. The specific problem of metal and composite layered structures was chosen due to its complexity and fit for the demonstration objectives.

The major purpose of this section is to determine important variables for modeling the damage tolerance of layered hybrid structure. In the absence of prior knowledge about variable importance, we compare the selection outcome of the proposed method with those of other alternatives. To ensure the quality of models used for feature selection, we compare the prediction accuracy of the models in addition to variable selection results. For more

effective comparison, 5-fold CV is used for this analysis. Since the other methods focus on the importance of "individual" variables, we extract a common set of important variables obtained from the 5-fold CV implementation for comparison purpose.

5.1 Finite Element Analysis

Numerical simulation using a validated finite element model (finite element analysis) is an efficient method to design and predict the damage tolerance of a layered hybrid structure for varying material properties. However, given the nearly unlimited choices of material combinations and stacking sequences, it is not possible to explore every possible design and optimize for all possible material and configuration parameters. Therefore, identification of the most influential parameters is necessary to limit the design and optimization parameter space to obtain a computationally tractable solution. The case study layered hybrid configuration is an aluminum plate with a co-cured bonded quasi-isotropic E-glass/epoxy composite overlay. The composite overlay consists of 8 layers of multiple lamina types (see Table 3) resulting in a high dimensional number of material parameters needed for characterization and input into the finite element model.

This layered multi-material model was loaded under four point bending to engage progressive damage in the structure through multiple damage mechanisms. Damage tolerance was evaluated by the total energy absorbed by the structure (an output parameter that is calculated during analysis and readily extracted using an automated approach). This model captures multiple, interacting damage mechanisms including the plastic deformation in aluminum, shear plasticity in each lamina, the intralaminar fracture of each lamina, delamination within the patch and disbond at the interface. Evaluation using this total damage energy provides a distinct and measurable result for the development of a reduced order predictive model and evaluation of influential parameters and their interactions.

A 3D high fidelity finite element model explicitly captures each layer in the hybrid structure as well as the interfaces (Heng and TerMaath, 2018). Each fabric layer (lamina) is explicitly modeled, and cohesive elements are included between each layer to capture delamination between plies. Each lamina is individually modeled with continuum shell elements (SC8R). A cohesive damage model is implemented for each lamina using a VUMAT user subroutine. Cohesive elements with a triangular traction-separation law are used to

	E-glass fabric	Fabrication style
1	Hexcel 7781	0°/90° Stain weave
2	Vectorply E-BX 1200	± 45° Stitch
3	Vectorply E-LT 1800	0°/90° Stitch
4	Vectorply E-BX 1200	± 45° Stitch
5	Vectorply E-BX 1200	± 45° Stitch
6	Vectorply E-LT 1800	0°/90° Stitch
7	Vectorply E-BX 1200	± 45° Stitch
8	Hexcel 7500	0°/90° Plain weave

Table 3: Composite patch stacking sequence

detect the interlaminar damage and are also included at the metal/composite interface to capture disbond between the metal and resin. The aluminum substrate is modeled with solid elements (C3D8R). Loading and support pins are modeled as rigid bodies to create the boundary and loading conditions. The numerical simulations are executed in the FE code ABAQUS. This physics-based model was validated under four point bend loading through physical testing (Fig. 2) .

To predict the total energy absorption of the layered hybrid structure, the 41 parameters characterizing the material properties needed for input into the finite element model for the multiple layers are used as the predictors (see Table 4 for detail). Latin hypercube sampling is applied to sample the parameter space based on the mean and standard deviation of the parameter ranges. For this case study, completing a single analysis generating a single data record takes an average of 3 hours depending on the parameter combination. Due to the computational time required to analyze the finite element model, we conducted 200 analyses producing a predictor matrix of size 200×41 and a response vector of size 200. With more analyses, it is possible to generate a larger data set, and this can further improve the prediction quality. However, from a practical perspective, we aim to develop a method that works well even with a small data set, so we compare alternatives based on this small data set.

5.2 Feature Selection Outcomes

In this section, we compare the proposed method with the same alternatives used in Section 4 and apply the implementation settings discussed in Section 4.2. For this particular data set, the proposed method creates and evaluates 3-dimensional subspaces, $k = 3$, and limits the maximum number of iterations to 10000 ($\binom{41}{3} = 10660$). Among 9 different hyperparameter level settings, this hard threshold for termination becomes active only for a single level combination, so the termination criterion described in Section 3.2 generally works well (on average, having 3154 iterations). The optimal hyperparameter values of the proposed method vary a little with each model learning set (five of such). Table 5 shows

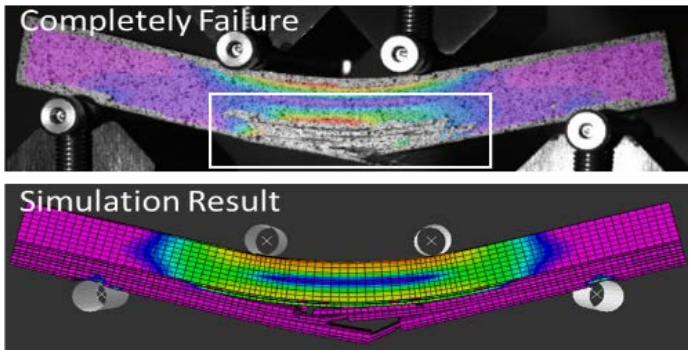

Figure 2: Comparison of experimental (top) and numerical results (bottom) at failure under four point bending

Variable	Corresponding parameter
x_1 (σ_y)	Yield Stress of Al-5456
x_2 (n)	Strain Hardening Exponent of Al-5456
x_3 (σ_{ss})	Nominal Stress First/Second Direction of Resin between Laminate plies
x_4 (E_{Al})	Young's Modulus of Al-5456
x_5 (P)	Power Term in Shear Hardening Equation for Laminates
x_6 (σ_{nn})	Nominal Stress Normal-only Mode of Resin between Laminate plies
x_7 (X_{7781})	Tensile strength of the Laminae Reinforced with Hexcel 7781
x_8 (G_{1200})	Intralaminar Fracture Toughness of Laminae Reinforced with EBX 1200
x_9 (ϵ^{pl}_{max})	Maximum Shear Plastic Strain for Laminates
x_{10} (G_{II})	Shear Mode Fracture Energy First/Second Direction of Resin between Laminate plies
x_{11} (α_{12})	Shear Damage Parameter for Laminates
x_{12} (E_{1800})	Young's Modulus of Laminae Reinforced with ELT 1800
x_{13} (G_I)	Normal Mode Fracture Energy of Resin between Laminate plies
x_{14} (X_{7500})	Tensile strength of the Laminae Reinforced with Hexcel 7500
x_{15} (σ_{nni})	Nominal Stress Normal-only Mode of Resin between metal/composite interface
x_{16} (v_{Al})	Poisson's Ratio of Al-5456
x_{17} (E_{7500})	Young's Modulus of Laminae Reinforced with Hexcel 7500
x_{18} (σ_{ssi})	Nominal Stress First/Second Direction of Resin between metal/composite interface
x_{19} (X_{1800})	Tensile strength of the Laminae Reinforced with ELT 1800
x_{20} ($B - K_i$)	Mixed Mode Behavior for Benzeggagh-Kenane of Resin between metal/composite interface
x_{21} (E_{1200})	Young's Modulus of Laminae Reinforced with EBX 1200
x_{22} (G_{1800})	Intralaminar Fracture Toughness of Laminae Reinforced with ELT 1800
x_{23} ($\bar{\sigma}_y$)	Effective Shear Yield Stress for Laminates
x_{24} (X_{12})	Shear Strength of Resin for all Lamina
x_{25} (B)	Strength Coefficient of Al-5456
x_{26} (v_{7500})	Poisson Ratio of Laminae Reinforced with Hexcel 7500
x_{27} (X_{1200})	Tensile strength of the Laminae Reinforced with EBX 1200
x_{28} (v_{1200})	Poisson Ratio of Laminae Reinforced with EBX 1200
x_{29} (v_{1800})	Poisson Ratio of Laminae Reinforced with ELT 1800
x_{30} (G_{7500})	Intralaminar Fracture Toughness of Laminae Reinforced with Hexcel 7500
x_{31} (E_{7781})	Young's Modulus of Laminae Reinforced with Hexcel 7781
x_{32} (v_{7781})	Poisson Ratio of Laminae Reinforced with Hexcel 7781
x_{33} (G_{7781})	Intralaminar Fracture Toughness of Laminae Reinforced with Hexcel 7781
x_{34} (G_{12})	Shear Modulus of Laminate for Laminates
x_{35} (d^{max}_{12})	Maximum Shear Damage for Laminates
x_{36} (C)	Coefficient in Shear Hardening Equation for Laminates
x_{37} (E_{nn})	Elastic Modulus of Resin between Laminate plies
x_{38} ($B - K$)	Mixed Mode Behavior for Benzeggagh-Kenane of Resin between Laminate plies
x_{39} (E_{nni})	Elastic Modulus of Resin between metal/composite interface
x_{40} (G_{Ii})	Normal Mode Fracture Energy of Resin between metal/composite interface
x_{41} (G_{IIi})	Shear Mode Fracture Energy First/Second Direction of Resin between metal/composite interface

Table 4: Variables for modeling damage tolerance of a hybrid metal structure (the Resin in this table is "Resin (M1002 with M2046 Hardener)") (Heng and TerMaath, 2018)

		Learn/test 1	Learn/test 2	Learn/test 3	Learn/test 4	Learn/test 5
A1	ϵ	0.01	0.01	0.01	0.01	0.01
	C	5	5	5	5	2
A2	ϵ	0.1	0.01	0.01	0.01	0.01
	C	5	5	5	5	2

Table 5: Optimal parameters selected for each of the five model learning/test data split

the optimal values selected based on the two GCV criterion suggested in Section 3.3. From the table, it is shown, in general, $\epsilon = 0.01$ and $C = 5$ are found optimal.

The optimal models based on the result in Table 5 are compared with other alternatives as shown in Table 6. It is noticed that, in general, linear models perform better than nonlinear models for the material damage tolerance prediction. This applies to both the average and standard deviation of the prediction errors. Among the linear methods, PLS regression produces the lowest average prediction error while the regularized methods (lasso and ridge) show comparably decent performance. Although LR attains the smallest standard deviation, its average prediction error is relatively higher than the other methods. For nonlinear methods, only SVM with a polynomial kernel and NN provide an average RMSE comparable to that of the linear counterpart. The two methods also show a similar level of the robustness of the estimator (through standard deviation). The proposed subspace-based method achieves the best performance among all the methods. The model based on A1 GCV criterion is comparable to the best linear methods, and the model based on A2 outperforms all other methods with respect to the average and standard deviation of the

Method	Average	Std. dev.
LR	12.61	1.05
Lasso	12.02	1.45
Ridge	11.98	1.20
PCR	12.21	1.64
PLS	11.89	1.15
RF	14.70	3.00
k-NN	15.27	3.51
SVM (RBF)	17.49	3.77
SVM (POLY)	12.01	1.20
NN	12.45	1.13
Subspace-SVM (A1)	11.99	1.12
Subspace-SVM (A2)	**11.64**	**0.94**

Table 6: Comparison of the average and standard deviation of RMSE's from 5-fold evaluation

RMSE's. This indicates that the proposed methods provide a model that is not only more accurate but also more robust and stable.

Although A1 applies the outcome of Theorem 1 exactly, it is based on the assumption of a squared loss function. As such, A1 is also an approximation of the exact GCV criterion. Since ϵ-insensitive loss function used in Eq. (5) penalizes errors out of the ϵ-tube proportionally to the distance from the tube, the calculation based on a squared loss function that more harshly penalizes larger errors may not guarantee the optimal performance. In A2, simplifying the hat matrix, i.e., dropping the recursive term of $(\mathbf{I} - \sum_{l=0}^{j-1} \mathbf{S}_l)$, removes dependency on other $\mathbf{S}_j$'s. This can reduce the variance of the estimator producing better results. In addition, from the computational perspective, A2 does not require storing and adding all the previous $\mathbf{S}_j$'s but just requires calculating the trace of $\mathbf{S}'_j$ for each subspace model estimation, so it is computationally more efficient than A1.

By construction, the randomized subspace model extracts critical subspaces. However, none of the other methods has this capability. To assess the variable selection capability of the randomized subspace method, we instead compare which individual variables are selected as an important predictor. As illustrated in Fig. 3, we first generate a set of all individual variables included in the selected subspaces for each 5-fold evaluation (used at the testing level), $\mathcal{X}_q$ for $q = 1, \ldots, 5$. Then, we extract common variables included in all five sets of important individual variables for comparison. Similarly, for all other methods, we find a set of important individual variables from each 5-fold evaluation and select the common variables from the sets, for the consistency of the comparison procedure.

The result of important individual variable selection is shown in Table 7. In the table, certain variables are identified as important variables by all methods while some are selected by only a few methods. For illustration purpose, we exclude all other variables that are not selected by any of the methods. According to the variable selection result, PCR and NN missed a variable (x_8 and x_3, respectively) that is chosen by all others. On the other hand, our method captures all variables that are found important by all the other methods,

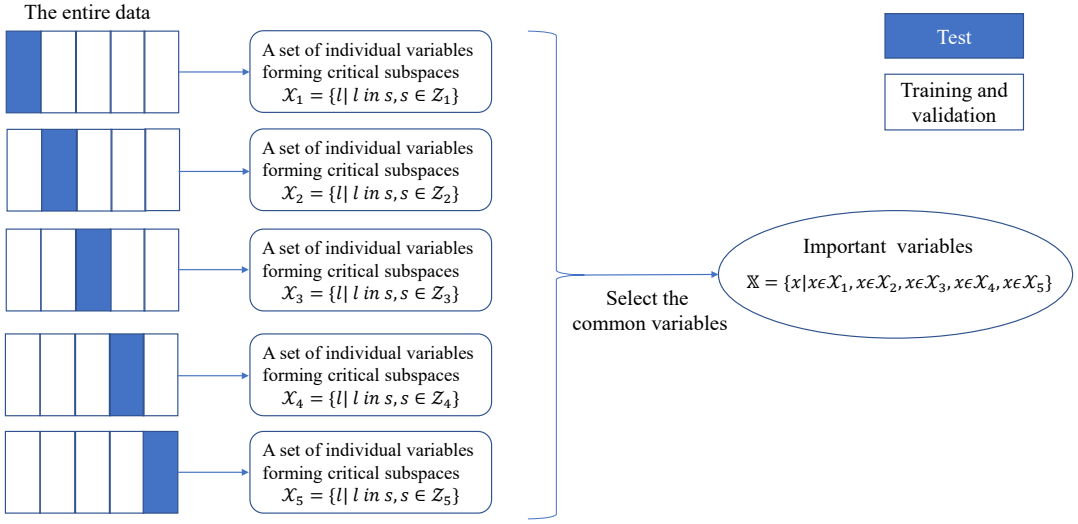

Figure 3: A procedure of selecting important individual variables

Method	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}
Linear	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓								
Lasso	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓						
Ridge	✓	✓	✓	✓	✓	✓	✓	✓			✓		✓	✓				
PCR	✓	✓	✓	✓			✓					✓	✓	✓	✓	✓		
PLS	✓	✓	✓	✓	✓		✓	✓		✓		✓						✓
RF	✓	✓	✓	✓	✓		✓	✓					✓			✓	✓	
k-NN	✓	✓	✓	✓			✓	✓				✓	✓	✓	✓		✓	
SVM (RBF)	✓	✓	✓	✓			✓	✓				✓	✓	✓	✓		✓	
SVM(POLY)	✓	✓	✓	✓			✓	✓				✓	✓	✓	✓		✓	
NN	✓	✓		✓	✓	✓	✓	✓				✓						
Our method	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓						

Table 7: Important variables selected by all alternatives; the alternatives with poor prediction errors are dimmed with gray color.

such as x_1, x_2, x_4 and x_7 as well as x_8 and x_3. Variables identified by the majority of the other methods (e.g., x_5 and x_{12}) are marked as important in our method. Certain variables rarely identified by the others (e.g., x_9 and x_{11}) are not included in the set of important variables from our method. In a few cases, our method selects a variable not chosen by the majority (such as x_6 and x_{10}) and does not select a variable voted by the majority (x_{13} and x_{14}). Note here that, however, according to Table 6, the prediction quality of RF, k-NN, and SVM with an RBF kernel is not quite acceptable. By excluding the variable selection results from these methods, our method is, in fact, capable of determining important and unimportant variables, of which result well aligns with the results of the remaining methods. In summary, this comparison result demonstrates that our method effectively extracts important variables.

With regard to the physical interpretation of these results, the total damage energy can be described as the sum of the energies contributed by 5 specific damage mechanisms (Arndt et al., 2022). Arndt et al. (2022) performed a comprehensive characterization and evaluation of the total damage energy behavior as well as the individual damage mechanisms. They determined that x_1, x_2, x_3, and x_4 were the most influential parameters by a considerable margin. For the loading case investigated, plastic deformation of the metal layer was significantly the largest contributor to the total energy with the corresponding significant parameters of x_1, x_2, and x_4 for this damage mechanism. Meanwhile, x_3 was a highly influential parameter for both the shear plasticity in the laminate and interlaminar fracture, which were the next two highest contributors to the total energy. Almost all methods accurately identified these parameters as significant. The next two most influential parameters from a physical perspective are x_5 and x_{10} which some of the methods, including ours, captured and others missed. x_{13} and x_{14} were not physically determined to contribute to any of the individual damage mechanics, however, were determined as significant by some of the other methods investigated. Our method correctly did not identify these two parameters as influential.

Discussion of the other parameters determined as significant but of less influence becomes more complex and dependent on the sampling method. As determined by Arndt

et al. (2022), not only does the total energy vary throughout the parameter space, but also the contributions of the damage mechanisms. Therefore, the significance of some parameters varies depending on the location in the parameter space according to which damage mechanisms govern at that location. For example, x_{11} is not one of the most significant parameters, but is identified as influential in both interlaminar fracture and interlaminar delamination and x_9 is not one of the most significant parameters, but is identified as influential in both shear plasticity in the laminate and interface disbond. These parameters could be significant to the total energy in local subspaces while not as influential globally. Alternatively, x_6 is not a significant parameter on its own from a physical perspective, but may contribute through parameter interaction, an effect requiring future investigation.

Besides the popular machine learning methods, we also compare our variable selection result with that of the elementary effects method (Heng and TerMaath, 2018) in Table 8. The result of the elementary effects method was derived from a data set generated by one-factor-at-a-time approach, and the method did not extract common variables from 5-fold CV and did not record the top 20 important variables (instead, reported the top 10 important variables). Albeit this is not a valid comparison as there are other aforementioned factors affecting the variable selection, we aim to compare our finding to the existing variable selection for this specific problem. In Table 8, there are four common variables selected by our method and the elementary effects method. The elementary effects method does not mark some variables found important by the majority of the machine learning-based methods, including x_2, x_3, x_7, and x_8, and it identifies some variables that are never selected by others, such as x_{24}, x_{31}, x_{33}, and x_{41}. Without knowing the true importance of the variables, we cannot draw a solid conclusion, but our methods result well aligns with the result of other alternatives whereas the existing selection from the elementary effects method is a bit far from the majority vote. Our method adds x_2, x_3, x_5, x_6, x_7, and x_8 as important variables and drops x_{11}, x_{17}, x_{24}, x_{31}, x_{33}, and x_{41} to and from the existing variable selection.

5.3 Critical Subspace Analysis

One of the major novelty of the randomized subspace modeling is the capability of extracting critical subspaces, equivalently, identifying important physical variables and interactions among the variables. In Section 5.2, the comparison result demonstrates the quality of the important variable selection of the proposed method. Note here that the result of the important (individual) variable selection was derived from critical subspaces identified in multiple model learning/testing data splits. This ensures the quality of the critical subspace selection to some extent.

To determine critical subspaces for the given data set, we used the entire data for model learning without leaving any separate data for testing. The final model is constructed by 11 distinct critical subspaces, as shown in Table 9. We observe that these subspaces contain

Method	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{17}	x_{24}	x_{31}	x_{33}	x_{41}
Elementary effects method	✓			✓						✓	✓	✓	✓	✓	✓	✓	✓
Our method	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓					

Table 8: Important variables selected by elementary effects method and our method

	Critical subspace	Important individual variables
1	x_{33}, x_4, x_{13}	x_4
2	x_{12}, x_{19}, x_{14}	x_{12}
3	x_{12}, x_1, x_{16}	x_1, x_{12}
4	x_8, x_3, x_{15}	x_3, x_8
5	x_{27}, x_7, x_1	x_1, x_7
6	x_{28}, x_{19}, x_{18}	/
7	x_{19}, x_{29}, x_2	x_2
8	x_5, x_{14}, x_{10}	x_5, x_{10}
9	x_{34}, x_3, x_{11}	x_3
10	x_{23}, x_4, x_{15}	x_4
11	x_9, x_2, x_3	x_2, x_3

Table 9: Critical subspace selected by our method

most of the important individual variables found in Table 7, except x_6, while each subspace excluding the sixth subspace includes at least one or two important individual variable(s). Assuming that an interaction formed by individually important variables is more significant than an interaction between others, we can argue that the interactions formed by (x_1, x_{12}), (x_3, x_8), (x_1, x_7), (x_5, x_{10}), and (x_2, x_3) are significantly important.

There are some variables that are not identified as important individual variables but included in the critical subspaces. These variables by themselves may not have significant impact on the damage tolerance, but their interactions with others could influence the response. In the current form of the proposed method, we cannot validate if any interaction of variables in a subspace is significant. We will investigate this aspect as one of future study. For a reference, if only one variable is added to the model at a time ($k = 1$), the prediction error is 12.28 with the standard error of 0.84. By ignoring the interactions formed by multiple variables, the prediction becomes worse. This supports, at least to the minimum extent, that some of the interactions we modeled through subspaces improve the prediction, and hence they are considered significant. Meanwhile, this determination of critical interactions improves our understanding of the complex engineering problem by identifying behaviors that may be outliers to our current level of understanding and characterization.

6. Conclusion

In this paper, we propose the randomized subspace modeling to alleviate challenges in high-dimensional data analysis and provide valuable insight about an underlying system by identifying important physical variables and interactions. The proposed method leverages an ensemble of multiple models derived from critical subspaces, reduced-dimensional physical spaces. The critical subspaces are generated by a randomized search and evaluated by a cross-validated selection criterion. With this structure, the proposed method shows its superiority over other alternatives in modeling a regression problem and identifying im-

portant variables. Specifically, from the analysis of hybrid material's damage tolerance, we derive the following conclusions:

1. Compared to other methods commonly used for variable selection (e.g., RF, NN, and Lasso), our method attains the lowest prediction error for this complex problem both in mean and standard deviation, demonstrating its competitiveness in high-dimensional prediction tasks.

2. The important variable selection result of the proposed method well aligns with the majority of other alternatives, verifying its variable selection capability. More importantly, our method identifies critical subspaces capturing not only important physical variables but also significant interactions, which is beneficial to experimental designs for broad engineering problems.

In the future, we plan to improve the randomized search process. This includes a weighted randomized search leveraging variable importance priors obtained from domain knowledge and a multi-step search process that first learns the rankings of variable importance and searches subspaces based on the rankings. Improving the randomized search can avoid adding insignificant variables in the selected subspaces. However, the inherent randomness can still let an insignificant variable involved in a subspace. This is mainly due to the imposition of fixed dimensionality of random subspaces. As such, we will also study how to identify and drop insignificant variables from the chosen subspaces allowing flexible dimensionality of the subspaces.

Acknowledgments

This research was supported in part by the United States Office of Naval Research (ONR) grant N00014-21-1-2041 under the direction of Dr. Paul Hess.

Appendix A. Proof of Theorem 1

Proof From Eq. (1), we define $\mathbf{S}_j\mathbf{y}$ for $j = 1, \ldots, J$ to represent $g_j(\mathbf{z}_{j,i})$ as a linear combination of y_i's so that we have

$$\hat{\mathbf{y}} = \mathbf{S}_1\mathbf{y} + \mathbf{S}_2\mathbf{y} + \cdots + \mathbf{S}_J\mathbf{y}. \tag{8}$$

Since each g_j estimates residuals from a model including up to the previous subspace-based model (g_{j-1}), we have $\tilde{y}_{j,i} = g_j(\mathbf{z}_{j,i}) + \tilde{\varepsilon}_{j,i}$ (as in Eq. (3)). For an SVR with a squared loss function, the estimation can be expressed as $\widehat{\tilde{\mathbf{y}}}_j = \mathbf{S}'_j\tilde{\mathbf{y}}_j = (\mathbf{K}_j + \lambda\mathbf{I})^{-1}\mathbf{K}_j\tilde{\mathbf{y}}_j$ (see Hastie et al. 2009) where $\widehat{\tilde{\mathbf{y}}}_j$ is the estimate of $\tilde{\mathbf{y}}_j$, a vector including $\tilde{y}_{j,i}$ for $\forall i$, and $\{\mathbf{K}_j\}_{i,i'} = K_j(\mathbf{z}_{j,i}, \mathbf{z}_{j,i'})$ for $i, i' = 1, \ldots, n$ for a given kernel $\mathbf{K}_j$. Since $\tilde{\mathbf{y}}_j = \mathbf{y} - \mathbf{S}_1\mathbf{y} - \mathbf{S}_2\mathbf{y} - \cdots - \mathbf{S}_{j-1}\mathbf{y}$,

$$\mathbf{S}_j\mathbf{y} := \widehat{\tilde{\mathbf{y}}}_j = \mathbf{S}'_j\tilde{\mathbf{y}}_j = \mathbf{S}'_j(\mathbf{y} - \mathbf{S}_1\mathbf{y} - \mathbf{S}_2\mathbf{y} - \cdots - \mathbf{S}_{j-1}\mathbf{y}) = \mathbf{S}'_j(\mathbf{I} - \sum_{l=1}^{j-1}\mathbf{S}_l)\mathbf{y} = \mathbf{S}'_j(\mathbf{I} - \sum_{l=0}^{j-1}\mathbf{S}_l)\mathbf{y}. \tag{9}$$

Now, for $j = 1$, the response to estimate is $\tilde{\mathbf{y}}_1 = \mathbf{y}$, so we have

$$\mathbf{S}_1\mathbf{y} := \widehat{\tilde{\mathbf{y}}}_1 = \mathbf{S}'_1\tilde{\mathbf{y}}_j = \mathbf{S}'_1\mathbf{y} = \mathbf{S}'_1(\mathbf{I} - \mathbf{S}_0)\mathbf{y}. \tag{10}$$

where $\mathbf{S}_0$ is an $n \times n$ zero matrix. Therefore, the result holds for $\forall j = 1, \ldots, J$. ■

Appendix B. Sensitivity Analysis

In this section, we show the sensitivity of the proposed method to some key parameters used for model construction. As the method relies on random generations of subspaces, we test the prediction capability with respect to multiple random seeds. In addition, we investigate the impact of selection and termination thresholds, i.e., η and τ, that are used for subspace selection and algorithm termination, respectively. Finally, we compare the goodness-of-fit of the proposed model for different dimensionality of subspaces, k.

To test the randomness, another 10 random seeds with an increment of 25000 are used to generate the randomized subspaces. The increment of 25000 is employed to avoid any duplicated sequences of subspace generation and fully investigate the effect of randomness; note, an exhaustive search requires evaluating $\binom{41}{3} = 10660$ subspaces for this data set. The result is shown in the Table 10. The mean of average prediction errors is 11.95 (by including the initial result) and the mean of standard deviations is 0.99 (by including the initial result). Although the average prediction errors increase a little on average, their mean is still better than other alternatives except PLS regression; please refer to Table 6. This indicates that the proposed method provides decent prediction in general regardless of the random seed. All individual prediction errors are comparable to those of the linear models. In fact, there are 2 cases (Tests 4 and 7) producing lower prediction errors than the one reported in Table 6 and another case (Test 3) that is comparable. The standard deviations of RMSEs are also comparable to the result shown in Table 6. There are three cases (Test 3, 7, and 8) achieving lower standard deviation.

Test	1	2	3	4	5	6	7	8	9	10
Avg.	12.39	12.01	11.7	11.41	11.78	12.25	11.41	12.27	12.41	12.25
Std. dev.	0.98	0.99	0.83	1.04	0.94	1.45	0.78	0.85	1.11	1.08

Table 10: The average and standard deviation of RMSE's from 5-fold evaluation using different random seeds

The result of various choices of selection and termination thresholds is given in Table 11. The proposed method is sensitive to the selection threshold η to some degree, and in the worst case, the prediction accuracy is worse compared to some nonlinear methods. As such, a careful selection of η is needed for a decent model. This also implies that the selection threshold plays an important role in controlling the uncertainty caused by the random generation of subspaces; without careful selection, the uncertainty can dominate the benefit of the random search. Although the impact could be less significant, the termination threshold τ plays a similar role. We recommend to start with some values between 1×10^{-4} and 1×10^{-6} and adjust the value depending the characteristics of a given data set.

We also vary the subspace dimension, k, and assess the prediction of models formed based on different dimensionality of subspaces. Table 12 shows the result. As the subspace dimension increases, the average prediction error decreases up to $k = 3$ and then increases. This clearly shows that the current selection of $k = 3$ provides the best prediction for this data set. The change in the dimensionality seems to have less impact compared to other parameters tested. However, please notice from Table 12, when $k = 1$, the method is

Selection			Termination		
η	Average	Std. dev.	τ	Average	Std. dev.
0.001	12.03	1.28	0.000001	11.79	0.87
0.005	12.34	1.16	0.000005	12.34	1.65
0.01	11.64	0.94	**0.000010**	11.64	0.94
0.05	12.24	0.82	0.000050	12.48	1.38
0.1	12.96	1.26	0.000100	12.03	1.14

Table 11: Sensitivity of the selection and termination thresholds

Subspace dimension (k)	Average	Std. dev.
1	12.28	0.84
2	12.23	1.5
3	11.64	0.94
4	11.94	1.37
5	12.06	1.24

Table 12: Sensitivity of the subspace dimensions

only capable of choosing individually important variables without any interaction between features, and this leads to the worst performance among all dimensionality tested.

References

Alex Alexandridis, Panagiotis Patrinos, Haralambos Sarimveis, and George Tsekouras. A two-stage evolutionary algorithm for variable selection in the development of rbf neural network models. *Chemometrics and Intelligent Laboratory Systems*, 75(2):149–162, 2005.

Corey Arndt, Bozhi Heng, Rochelle Butler, and Stephanie TerMaath. Nondeterministic parameter space characterization of the damage tolerance of a composite/metal structure, 2022. URL `https://arxiv.org/abs/2210.14054`.

Mariette Awad and Rahul Khanna. Support vector regression. In *Efficient Learning Machines*, pages 67–80. Apress, Berkeley, CA, 2015.

Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.

Anna C Belkina, Christopher O Ciccolella, Rina Anno, Richard Halpert, Josef Spidlen, and Jennifer E Snyder-Cappione. Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. *Nature Communications*, 10(1):1–12, 2019.

Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

Rung-Ching Chen, Christine Dewi, Su-Wen Huang, and Rezzy Eko Caraka. Selecting critical features for data classification based on machine learning methods. *Journal of Big Data*, 7(1):1–26, 2020.

Sang Wook Choi, Changkyu Lee, Jong-Min Lee, Jin Hyun Park, and In-Beum Lee. Fault detection and identification of nonlinear processes based on kernel pca. *Chemometrics and Intelligent Laboratory Systems*, 75(1):55–67, 2005.

Frauke Degenhardt, Stephan Seifert, and Silke Szymczak. Evaluation of variable selection methods for random forests and omics data sets. *Briefings in Bioinformatics*, 20(2): 492–503, 2019.

Lakshmi Padmaja Dhyaram and B Vishnuvardhan. Random subset feature selection for classification. *International Journal of Advanced Research in Computer Science*, 9(2), 2018.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2019. URL `http://archive.ics.uci.edu/ml`. Last accessed 16 June 2022.

Simon Fong, Yan Zhuang, Rui Tang, Xin-She Yang, and Suash Deb. Selecting optimal feature set in high-dimensional data by swarm search. *Journal of Applied Mathematics*, 2013, 2013.

Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. Elsevier, San Diego, CA, 2013.

Andrej Gisbrecht, Alexander Schulz, and Barbara Hammer. Parametric nonlinear dimensionality reduction using kernel t-sne. *Neurocomputing*, 147:71–82, 2015.

Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

Baptiste Gregorutti, Bertrand Michel, and Philippe Saint-Pierre. Correlation and variable importance in random forests. *Statistics and Computing*, 27(3):659–678, 2017.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, New York, NY, 2009.

Bozhi Heng and Stephanie C TerMaath. Prediction of damage tolerance in metallic structure repaired with a co-cured composite patch. In *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0225, Kissimmee, FL, 2018.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Tommy Johnsson. A procedure for stepwise regression analysis. *Statistical Papers*, 33(1): 21–29, 1992.

Miron Bartosz Kursa. Robustness of random forest-based gene selection methods. *BMC Bioinformatics*, 15(1):1–8, 2014.

John A Lee and Michel Verleysen. *Nonlinear Dimensionality Reduction*. Springer Science & Business Media, New York, NY, 2007.

Yingzhu Li, Xunpeng Shi, and Lixia Yao. Evaluating energy security of resource-poor economies: A modified principle component analysis approach. *Energy Economics*, 58: 211–221, 2016.

Jeremiah Zhe Liu. Variable selection with rigorous uncertainty quantification using Bayesian deep neural networks. In *Bayesian Deep Learning Workshop at NeurIPS*, Vancouver, Canada, 2019.

Gilles Louppe, Louis Wehenkel, Antonio Sutera, and Pierre Geurts. Understanding variable importances in forests of randomized trees. *Advances in Neural Information Processing Systems*, 26, 2013.

Thomas Marill and D Green. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, 9(1):11–17, 1963.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. URL https://arxiv.org/abs/1802.03426.

Piyushkumar A Mundra and Jagath C Rajapakse. Svm-rfe with mrmr filter for gene selection. *IEEE Transactions on Nanobioscience*, 9(1):31–37, 2009.

Julian D Olden, Michael K Joy, and Russell G Death. An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. *Ecological Modelling*, 178(3-4):389–397, 2004.

Karl Pearson. Liii. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

Pavel Pudil, Jana Novovičová, and Josef Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, 1994.

G Thippa Reddy, M Praveen Kumar Reddy, Kuruva Lakshmanna, Rajesh Kaluri, Dharmendra Singh Rajput, Gautam Srivastava, and Thar Baker. Analysis of dimensionality reduction techniques on big data. *IEEE Access*, 8:54776–54788, 2020.

Nasir Saeed, Haewoon Nam, Tareq Y Al-Naffouri, and Mohamed-Slim Alouini. A state-of-the-art survey on multidimensional scaling-based localization techniques. *IEEE Communications Surveys & Tutorials*, 21(4):3565–3583, 2019.

Fadi Salo, Ali Bou Nassif, and Aleksander Essex. Dimensionality reduction with ig-pca and ensemble classifier for network intrusion detection. *Computer Networks*, 148:164–175, 2019.

Lawrence K Saul, Kilian Q Weinberger, Fei Sha, Jihun Ham, and Daniel D Lee. Spectral methods for dimensionality reduction. *Semi-supervised Learning*, 3, 2006.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

Petr Somol, Pavel Pudil, Jana Novovičová, and Pavel Paclık. Adaptive floating search methods in feature selection. *Pattern Recognition Letters*, 20(11-13):1157–1163, 1999.

Nicolas Sompairac, Petr V Nazarov, Urszula Czerwinska, Laura Cantini, Anne Biton, Askhat Molkenov, Zhaxybay Zhumadilov, Emmanuel Barillot, Francois Radvanyi, Alexander Gorban, et al. Independent component analysis for unraveling the complexity of cancer omics datasets. *International Journal of Molecular Sciences*, 20(18):4414, 2019.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Stephanie C TerMaath. Probabilistic multi-scale damage tolerance modeling of composite patches for naval aluminum alloys. Technical Report DTIC AD1074284, Department of Mechanical, Aerospace, and Biomedical Engineering, University of Tennessee, Knoxville, Tennessee, 2018.

Sebastian B Thrun, Jerzy W Bala, Eric Bloedorn, Ivan Bratko, Bojan Cestnik, John Cheng, Kenneth A De Jong, Saso Dzeroski, Douglas H Fisher, Scott E Fahlman, et al. The monk's problems: A performance comparison of different learning algorithms. Technical report, Department of Computer Science, The Carnegie Mellon University, Pittsburgh, Pennsylvania, 1991.

Kari Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, 2003.

Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:1–41, 2009.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pages 1954–1963, Lille, France, 2015a.

Pengfei Wei, Zhenzhou Lu, and Jingwen Song. Variable importance analysis: A comprehensive review. *Reliability Engineering & System Safety*, 142:399–432, 2015b.

A Wayne Whitney. A direct method of nonparametric measurement selection. *IEEE Transactions on Computers*, 100(9):1100–1103, 1971.

Zhou Xu, Jin Liu, Xiapu Luo, Zijiang Yang, Yifeng Zhang, Peipei Yuan, Yutian Tang, and Tao Zhang. Software defect prediction based on kernel pca and weighted extreme learning machine. *Information and Software Technology*, 106:182–200, 2019.

Bing Xue, Mengjie Zhang, Will N Browne, and Xin Yao. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, 20(4):606–626, 2015.

Yuling Yao, Aki Vehtari, Daniel Simpson, and Andrew Gelman. Using stacking to average Bayesian predictive distributions (with discussion). *Bayesian Analysis*, 13(3):917–1007, 2018.

Zhengze Zhou and Giles Hooker. Unbiased measurement of feature importance in tree-based methods. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(2):1–21, 2021.

Liao Zhu. *The Adaptive Multi-Factor Model and the Financial Market*. PhD thesis, Cornell University, Ithaca, NY, 2020.

Liao Zhu, Sumanta Basu, Robert A Jarrow, and Martin T Wells. High-dimensional estimation, basis assets, and the adaptive multi-factor model. *Quarterly Journal of Finance*, 10(04):2050017, 2020.

Liao Zhu, Robert A Jarrow, and Martin T Wells. Time-invariance coefficients tests with the adaptive multi-factor model. *The Quarterly Journal of Finance*, 11(04):2150019, 2021.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.