Classification-based Optimization and Reliability Assessment Using LS-OPT®

Anirban Basudhar[1], Imtiaz Gandikota[1], Nielen Stander[1], Åke Svedin[2], Christoffer Belestam[2], Katharina Witowski[3]

[1]Livermore Software Technology Corporation, Livermore CA, USA
[2] DYNAmore Nordic, Sweden
[3] DYNAmore GmbH, Germany

1 Introduction

Simulation-based design optimization and probabilistic analysis involve repeated calls to potentially expensive (e.g. crashworthiness analysis) design alternative or function evaluations. To avoid the high cost (or computational time) associated with repeated expensive evaluations, the actual function evaluations are substituted with evaluations based on metamodels. Metamodels, trained based on relatively few actual function evaluations, provide an approximation of the system responses and thus act as surrogate models.

Considerable research has been done in the field of metamodel-based design optimization and analysis [1]. Various types of metamodels can be trained based on different criteria for the "best" fit, several of which are also available in LS-OPT [2]. Different sampling schemes have also been developed to globally or locally enhance metamodel accuracy, based on specific applications such as deterministic single-objective design optimization, multi-objective optimization, reliability analysis or reliability-based design optimization [3-6]. For most types of problems, metamodel-based methods have been shown to perform efficiently and accurately. However, the use of metamodels is hampered for certain systems that have discontinuous or binary responses, e.g. buckling, on-off contact, hidden constraints etc [7, 8]. The approximation of response values becomes difficult for such systems.

This paper presents an alternative method, based on the classification of response values, which does not require response approximation [7, 9]. As a result, it is unaffected by the presence of binary or discontinuous responses and provides a straightforward way to solve such problems. The principal idea is to classify the training samples into two classes, using a threshold value or a clustering method, and then construct an "optimal" decision boundary in the design space that separates the samples belonging to the different classes. Thus, this decision boundary can be used as the limit-state in reliability analysis [7] or to constrain feasible design alternatives in optimization [10].

The use of classification methods is quite common in pattern recognition and decision making [11-13], but their use in engineering design is relatively new and has only gained attention over the last decade. A machine learning technique known as support vector machine (SVM) [14] has received special attention due to its ability to minimize generalization error and to define highly nonlinear boundaries needed to solve design problems in the general case. It was first used for reliability assessment in [15] and for deterministic or reliability-based optimization in [7]. Adaptive sampling techniques have also been developed based on the specific application (reliability analysis or optimization) [10, 15-18]. More recently, another classification method known as random forest classification has also gained attention [8]. However, no comprehensive study has been performed to assess the relative efficiency and accuracy of different classification methods. As in the case of metamodels, it is likely that the relative performance of different classifiers is also problem specific. Irrespective of the type of classifier, the basic principles of their application to engineering design remain the same.

The aim of this paper is to give an overview of different applications and advantages of classification in engineering design. A brief overview of the newly implemented MATLAB interface in LS-OPT is also presented. This interface can be used to integrate LS-OPT with open source MATLAB classification codes, but LS-OPT will have the classification algorithms implemented within itself in the near future. A brief overview of the proposed new "Classifier" entity in LS-OPT is also presented. Additionally, a major part of the paper is dedicated to a very recent method for classification-based multiobjective optimization (MOO) [19] developed by the principal author. This method has been compared to existing MOO methods, such as NSGAII [20] and Pareto Domain Reduction (PDR) [21], using several examples up to thirty variables. The SVM classification-based method involves a radical shift from current MOO methods, and is shown to perform significantly better.

2 Basic principle of clasification-based design

In both design optimization and reliability assessment one of the main tasks is the demarcation between acceptable (feasible/safe) and unacceptable (infeasible/failed) designs. In optimization, the optimum design is located in the feasible space. Similarly, in reliability assessment, the failed samples contribute to the failure probabiliy. If the boundary separating acceptable and unacceptable regions of the design space is available analytically in terms of the design variables, reliability assessment and optimization become relatively straightforward. However, in general such a boundary is not available. Instead, only the responses corresponding to specific designs are available. In metamodel-based approaches the response values at these specific points in the design space are used to construct analytical response approximations at any general design. These approximations are then used to demarcate the acceptable and unacceptable space based on threshold values. However, a different approach is used in classification based design. Classification methods only require pass/fail information at a few specified samples that are used for training. This information is readily available using simulations at these samples even if the responses are binary or discontunuous. The decision boundary is constructed as the classifier that optimally separates the acceptable and unacceptable training samples. The difference between metamodel-based and classification-based methods to determine acceptability of any general design alternative is shown in Figure 1. The classification-based method takes a decision directly based on the position of the new sample in the design space whereas in the metamodel-based method, the decision is taken based on the corresponding predicted response value and threshold. As the decision-making using a trained classifier is straightforward and cheap, the decision boundary can be used as an optimization constraint or for reliability assessment.

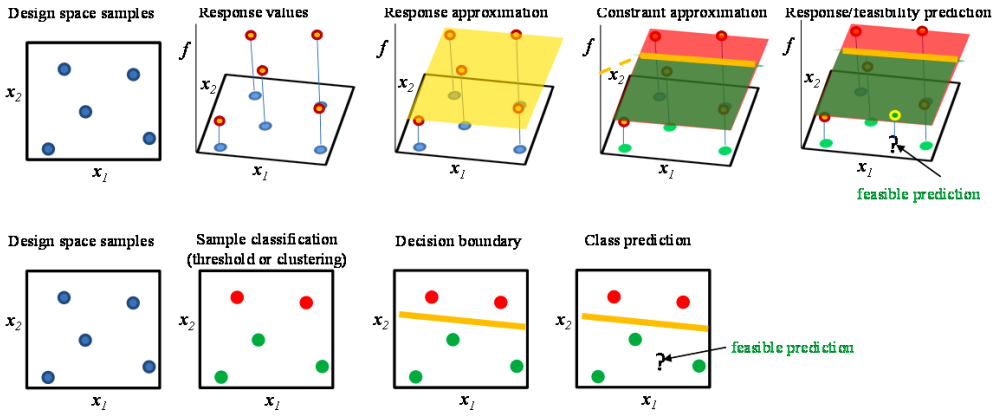

Figure 1 Summary of basic classification method (bottom) and comparison to metamodeling (top).

A prominent method to cnstruct the decision boundaries is SVM. An SVM boundary is obtained as s(x)=0, where s(x) is given in Eq. (1).

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

Here, $y_i = \pm 1$ (e.g. red vs green) is the class label, α_i is the Lagrange multiplier for i^{th} sample and b is the bias. A kernel K maps the design space and the feature space (a high-dimensional space consisting of basis functions as the variables, where the classifier is linear). In this work, a Gaussian kernel is used to construct SVM boundaries ($s(\mathbf{x}) = 0$). Spread of the kernel is assigned the largest value without any training misclassification. An SVM constructs the optimal boundary that maximizes the margin between two sample classes (± 1) in a feature space. The boundary is linear in the feature space, but can be highly non-linear in the design space.

The main advantages of using the classification-based method are as follows:

1. As evident in Figure 1, the classification-based method does not require the actual response values at the training samples; instead it only requires the class ± 1 of the samples. As a result, it can be applied to problems with only binary pass/fail information available (Figure 8).
2. As the response values are not approximate and are instead classified, the method is unaffected by the presence of discontinuities [7, 22].
3. Different failure modes can be represented by a single classifier. This feature can be used to devise a mechanism in which only a few failure modes need to be evaluated at a sample to obtain the class label, thus increasing the efficiency [10].

3 Classification-based design using a MATLAB stage in LS-OPT

The aim of this section is two-fold: to introduce the new MATLAB solver interface in LS-OPT, and to show how it can be used currently to perform classification-based design within the LS-OPT framework. In the future, classification method(s) will be implemented in LS-OPT to eliminate the need for the MATLAB stage for this particular purpose. First, a simple single stage optimization setup with a MATLAB stage type is presented. Then the process flow for classification-based design is presented.

The global problem setup and the MATLAB stage setup for a simple single iteration optimization are shown in Figure 2. An output file is required for the MATLAB stage that provides a template for the stage histories and responses. The formatting for this file is the same as for METAPost. Specification of the template output file automatically populates the response and history panels of the GUI. The input file for the MATLAB stage ends with ".m" and consists of the variable definitions and the code for response/history calculation. The variables are defined using the "input" function in MATLAB. To define a variable "*x1*", the MATLAB stage input file contains a command that resembles "`x1 = input('arbitrary text');`". The MATLAB input contains an appropriate try-catch statement to indicate the termination status (`Normal` or `Error`). Also, it needs to dump the responses and histories in the METAPost format (details in the LS-OPT manual [2]).

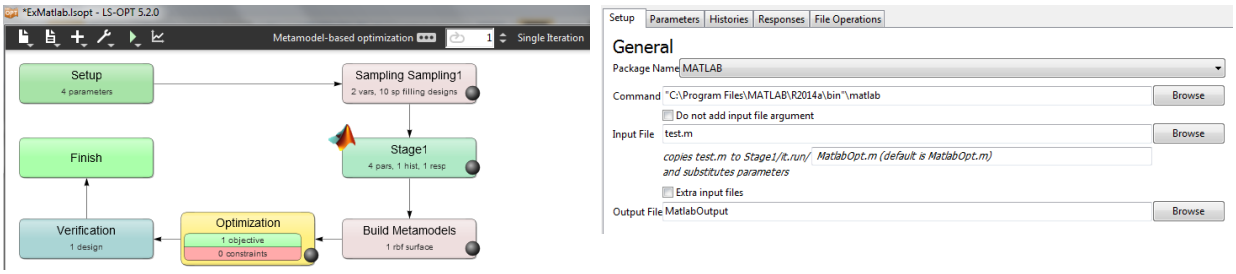

Figure 2 Single iteration optimization flow with MATLAB stage (left) and stage setup dialog (right).

Figure 3 shows the multilevel setup for classification-based design using a MATLAB stage. The outer level has an LS-OPT stage followed by a MATLAB stage, and a dummy sample and composite. The inner level DOE task has a user-defined sampling file (overwritten by the MATLAB stage in every iteration) and an LS-DYNA stage. The inputs to the LS-DYNA stage are the user-defined variable values and the outputs are responses and/or histories extracted from the LS-DYNA databases. These responses and their approximations are transferred to the MATLAB stage as inputs to train the classifier (using available SVM codes, e.g. [23]) and perform the optimization. The output of the MATLAB stage is either the final optimum solution or a set of new samples to overwrite the LS-DYNA sample input. In this setup, LS-OPT is used for LS-DYNA response extraction and metamodel construction. The optimization or reliability assessment is performed within the MATLAB stage using the classifier and the metamodel expressions exported by LS-OPT. In the future, these will be integrated in LS-OPT through the implementation of a new *classifier* entity that will share some of the characteristics of composites and metamodels. It is noteworthy that the multilevel setup consisting of a MATLAB stage for classification and optimization has been presented with the intent to demonstrate a possible use of the stage for sampling guidance. A simpler approach to this particular problem may be to call LS-OPT from MATLAB to extract the LS-DYNA responses (instead of the other way around).

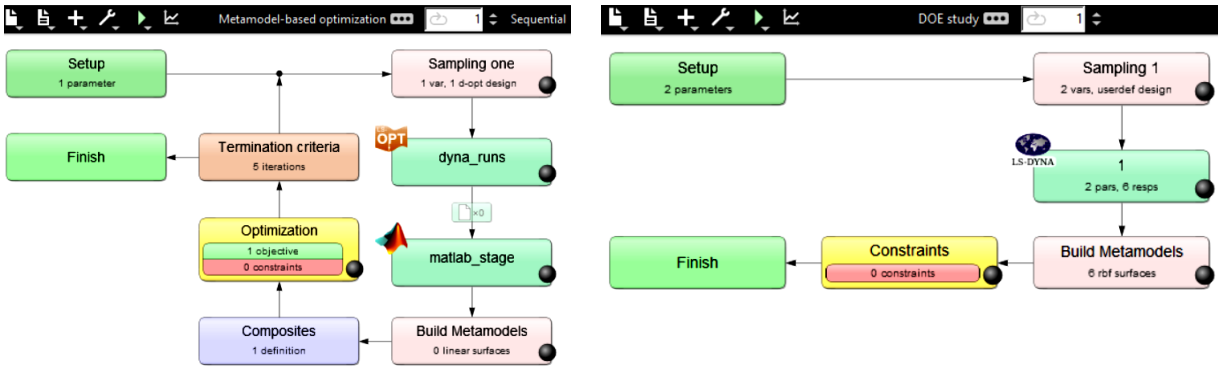

Figure 3 Sample selection and optimization (e.g. classification-based) using MATLAB stage. Outer level for iterative call to MATLAB (left) and inner level for response extraction at the samples written by MATLAB (right).

4 Application of Classification to MOO (Adaptive Explicit Multi-objective Optimization)

This section presents a newly developed method for MOO based on the definition of an explicit decision boundary in the design space that delineates the Pareto-optimal regions. The method is referred to as Adaptive Explicit Multiobjective Optimization (AEMOO). Two variants of the AEMOO method are presented. In Section 4.1 the basic idea of classification-based MOO is presented. In Section 4.2 a classifier-assisted direct optimization method is presented. A second method that utilizes classification as well as metamodel approximation is presented in Section 4.3.

4.1 Classification approach to MOO (Dynamic Classifiers)

In general, MOO problems consist of several objectives that are maximized/minimized simultaneously and are conflicting in nature. As a result, the solution of a MOO problem is typically a set of Pareto-optimal points instead of a single solution. A part of the design space is Pareto-optimal whereas other regions are dominated. Thus, MOO can be seen as a binary classification problem in which the two classes are dominated (-1 class) and non-dominated or ND (+1 class) (Figure 4). Once a classifier (SVM in this work) separating the dominated and non-dominated samples is trained, Pareto-optimality of any design is determined using a single classifier evaluation, in contrast with all existing methods.

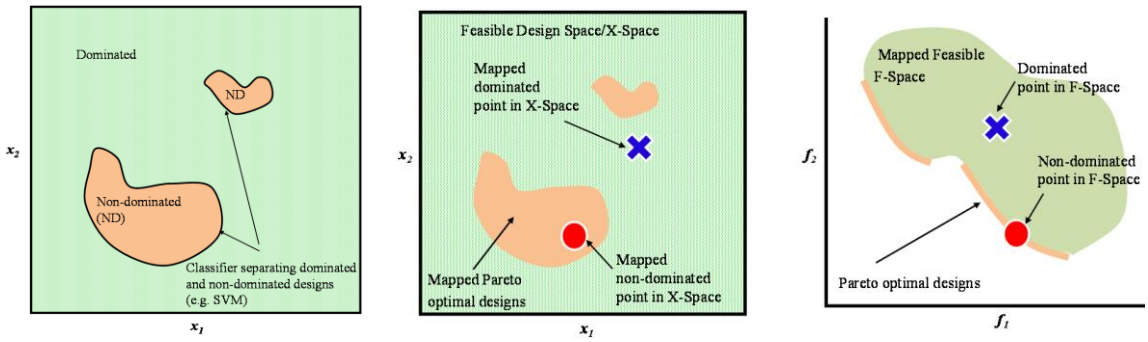

Figure 4: Boundary defining non-dominated regions in design space (left). Design-Objective space Mapping (right).

In order to train an SVM, each training sample's class needs to be assigned prior to the construction of the decision boundary. An issue in using classification for MOO is that although it is known that dominated samples (class -1) cannot be Pareto-optimal, the opposite is not true; being non-dominated among current samples isn't sufficient to be Pareto-optimal. However, the +1 samples represent Pareto-optimal designs if the data is sufficient. A decision boundary obtained by assigning +1 class to the current non-dominated samples represents an estimate of the Pareto-optimal front, and is refined adaptively. As points are added, samples may switch from +1 to -1 as non-dominated samples may be dominated by newer samples until convergence. As class definition of existing samples can change during the course of AEMOO, the classifier is referred to as *dynamic*. The classification based AEMOO method has several advantages.

1. Explicit definition of the non-dominated region facilitates implementation of an efficient sampling scheme.
2. It facilitates efficient real time Pareto optimality decisions.
3. It uses information in both design and objective spaces to enhance its efficiency and robustness.
4. The classification approach allows the handling of binary and discontinuous constraint functions.
5. As the non-dominated region is explicitly defined in the design space, AEMOO is not affected much by the number of objectives.

4.2 Direct AEMOO Method

Direct AEMOO is based on SVM classification only and does not approximate the function values. It favours the explicitly defined SVM-based non-dominated regions for sample selection to avoid waste of samples and increase the efficiency. The sign of SVM value s(**x**) determines whether a sample is non-dominated or not, and is straightforward to determine.

A fraction of the total *n* samples are selected within the s(**x**) > 0 non-dominated regions of the design space at each iteration (n_{svm} samples). Sampling important regions of the design space allows a faster increase in the SVM accuracy, as sampling based only on the objective space can lead to clustering of samples in the design space, where the SVM is constructed. Several criteria are considered to identify important regions with the potential to improve the SVM and the Pareto front (Figure 5):

1. Sparseness in the design space based on Euclidean distances
2. Sparseness in the objective space based on Euclidean distances
3. Probability of being non-dominated based on the SVM
4. Ranking based on non-domination criterion

Maximizing the value of SVM, one of the sampling criteria, is equivalent to maximizing the probability of locating a non-dominated sample [10, 14]. Additionally, one generation of NSGAII is also used to select samples, first within the s(**x**) > 0 regions and then using an unconstrained formulation. Using NSGAII-based samples, the algorithm ensures that the effects of sparseness in the objective function space and genetic operator-based evolution are also accounted for. In order to ensure a global search, a small fraction of samples is added based on maximum minimum distance in the entire unconstrained design space. Such samples are not expected to provide an efficient sampling scheme, and are therefore optional, but guarantee the location of the complete Pareto front when allowed to run sufficiently long.

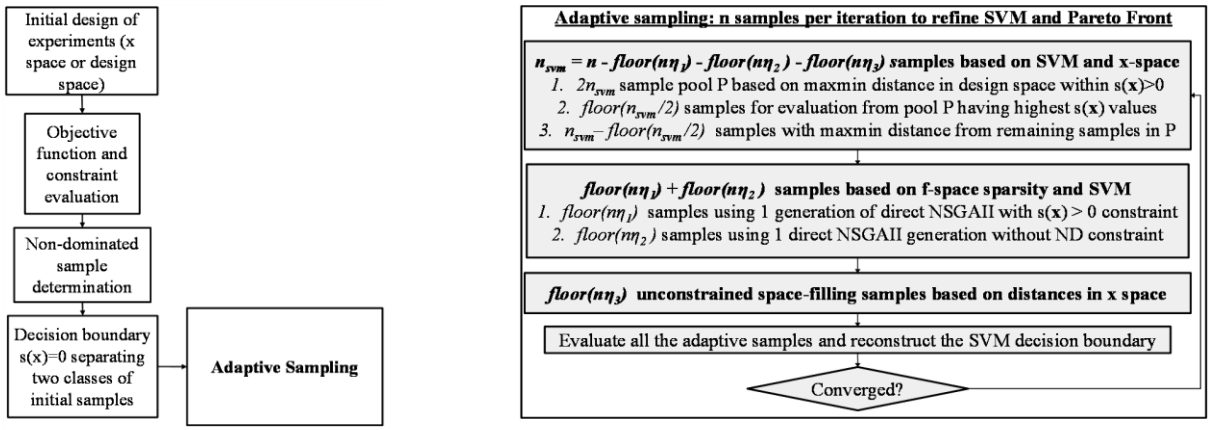

Figure 5: Summary of direct AEMOO method (left) and sampling scheme (right)

4.3 Metamodel-assisted AEMOO Method

In this approach, metamodel-based approximation and SVM are used together. The basic idea is the same as direct AEMOO - to consider non-dominated ranking along with sample sparseness in both the design (x) and objective (f) spaces. However, the single generation of direct NSGA-II samples is replaced with converged predicted Pareto-optimal samples obtained using metamodel-based NSGAII. Metamodel approximation and the SVM-based classification serve as complementary approaches that help in enhancing accuracy by accounting for the distribution of samples in both spaces. The methodology is shown in Figure 6.

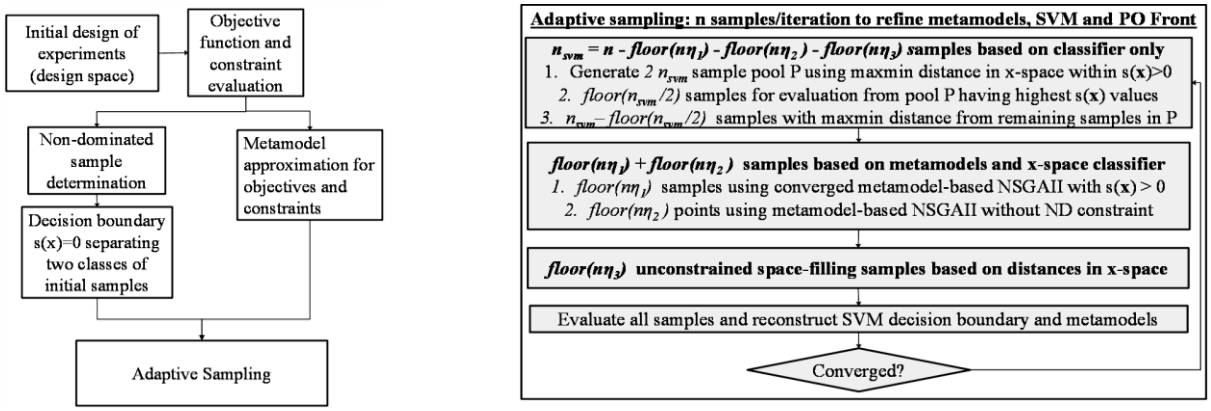

Figure 6: Summary of metamodel-assisted AEMOO method (left) and sampling scheme (right)

5 Examples

Several examples are presented to display the main capabilities of classification-based method. First, a binary problem is presented in Section 5.1. The aim of this example is to demonstrate the basics of classification and its advantages for binary and discontinuous responses. Several MOO examples are then presented to validate the efficacy and efficiency of AEMOO. A 10 variable analytical example using direct AEMOO is presented, followed by three 30 variable examples using metamodel-assisted AEMOO. The efficiency is compared to existing methods PDR and NSGAII. Finally, AEMOO is used for tolerance-based MOO of a truck [24]. The sample type fractions are η_1= η_2=0.2 and η_3=0.1 (Figure 5 and 6). Unless mentioned, *floor(1.5*(m+1))+1* samples are used per iteration, *m* being the number of variables. Cross-validated radial basis function metamodels, as implemented in LS-OPT, have been used for function approximations, but other metamodels can also be used. For examples 2-4, one of the objectives is $f_1 = x_1$. The second objective f_2 is provided with the individual examples.

5.1 Example 1. Plate failure due to impact (binary response)

This example consists of a ball impacting a plate modelled with solid elements and the MAT24 material card in LS-DYNA (Figure 7). The plate elements erode upon reaching a specified limiting plastic strain of 0.5. Upon reaching the limit state, an element fails and undergoes an instantaneous drop in strain to zero value. The exact limit state may or may not be captured for a particular simulation in the LS-DYNA output databases, depending on the output intervals. As a result, only pass/fail information is available in the `messag` file. This example is studied with only two variables, plate thickness and the yield stress, for simplification of the visual interpretation. An experimental design consisting of 50 space filing samples is generated. The corresponding binary states represented by +1 (no erosion) and -1 (element erosion) are shown in Figure 7. In Figure 8, the metamodel-based approach is compared to the classification-based approach. Metamodels have problems while approximating such binary or discontinuous responses and are sensitive to the selection of arbitrary numeric labels. However, this is a very simple problem from a classification perspective. The SVM boundary can be used for optimization or for reliability assessment.

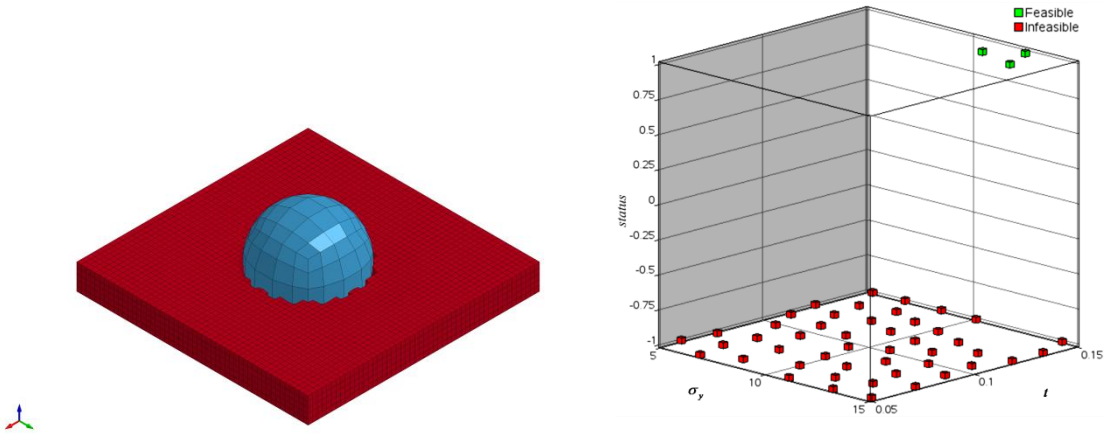

Figure 7: Element erosion upon impact on plate (left). Binary response (right). -1 shows erosion.

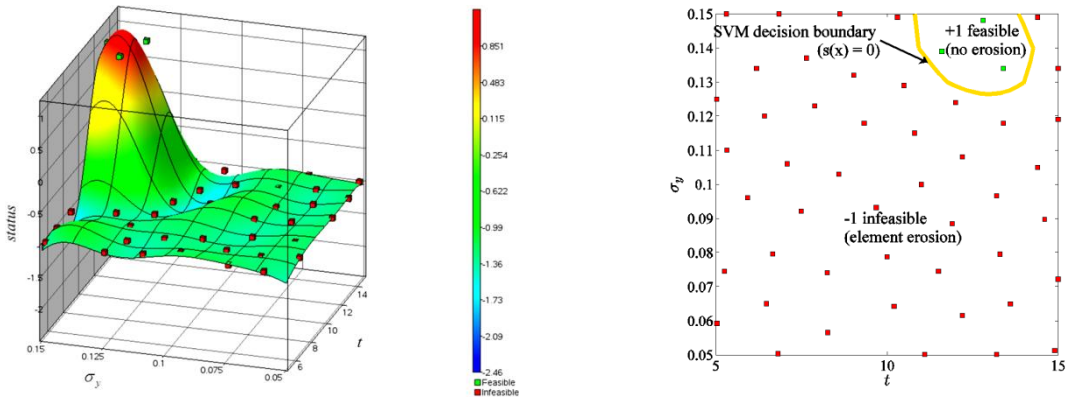

Figure 8: Fitting metamodel to binary response (left). SVM classification of the designs (right)

5.2 Example 2. Example with ten variables and two objectives - ZDT3 (Direct AEMOO)

This example has 10 variables (m = 10) and 2 objectives. The second objective f_2 is:

$$f_2 = r(\mathbf{x})h(f_1(\mathbf{x})r(\mathbf{x})), \text{ where } r(\mathbf{x}) = 1 + \frac{9}{m-1}\sum_{i=2}^{m} x_i \text{ , } h(\mathbf{x}) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{r(\mathbf{x})}} - \left(\frac{f_1(\mathbf{x})}{r(\mathbf{x})}\right)\sin(10\pi f_1(\mathbf{x})) \quad (2)$$

The Pareto fronts at successive iterations are plotted in Figure 9. The front at iteration 125 (2250 points) is quite close to the actual one, and shows that AEMOO can locate disjoint Pareto-optimal fronts using only classification. NSGAII completely missed one region among five on the front. This can be attributed to sampling based on the f-space only without considering design space sparseness, unlike in AEMOO.

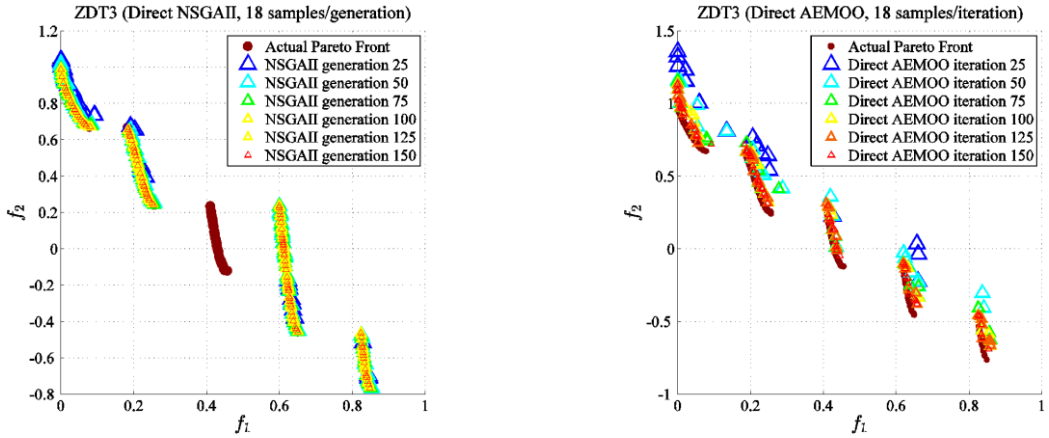

Figure 9: Results for Example 1. Direct NSGAII (left) and direct AEMOO (right).

5.3 Example 3. ZDT1 with 30 variables and 2 objectives (metamodel-assisted AEMOO)

This problem consists of two objectives f_1 and f_2. The second objective is:

$$f_2 = r(\mathbf{x})h(f_1(\mathbf{x})r(\mathbf{x})), \text{ where } r(\mathbf{x}) = 1 + \frac{9}{m-1}\sum_{i=2}^{m} x_i \text{ , } h(\mathbf{x}) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{r(\mathbf{x})}} \quad (3)$$

Optimization results are shown in Figure 10 using trade off plots at different iterations at intervals of 10. The results shown are the evaluated Pareto optimal points. The proposed AEMOO method is able to locate the entire spread of Pareto front at the 10th iteration itself (470 samples), after which it adds diversity. The samples on the front are very well diversified by the 20th iteration. In comparison, performance of direct NSGAII is much slower and it takes 40-50 generations (1920-2400 samples) to obtain a diversified front. Even at 50th generation, the Pareto front using NSGAII is not as accurate as the 10th iteration of AEMOO. PDR performs more efficiently than direct NSGAII, but still takes 20-30 iterations (940-1410 samples) to obtain a well diversified accurate front.

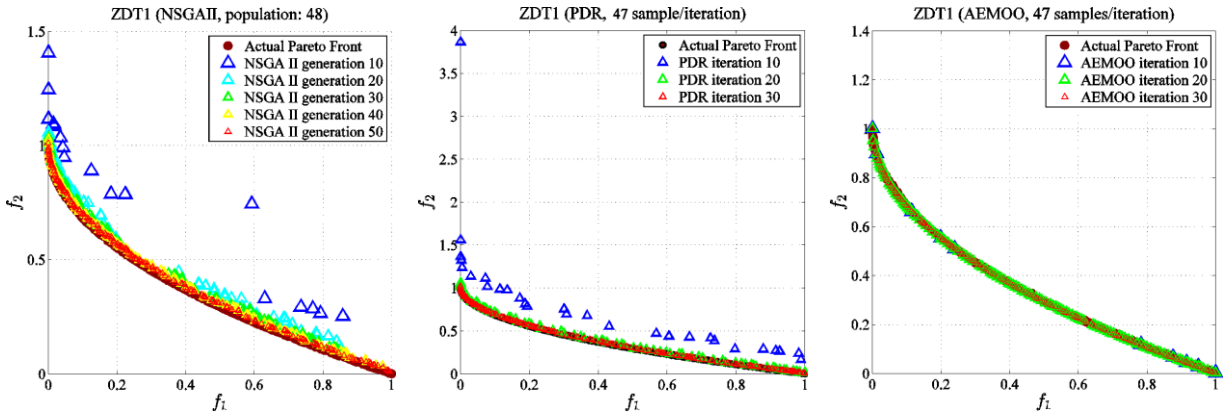

Figure 10: Example 2. Direct NSGAII (left), PDR (center) and metamodel-assisted AEMOO (right).

5.4 Example 4. ZDT2 with 30 variables and 2 objectives (Metamodel-assisted AEMOO)

This problem consists of 30 variables (m = 30) and two objective functions. The second objective is:

$$f_2 = r(\mathbf{x})h(f_1(\mathbf{x})r(\mathbf{x})), \text{ where } \quad r(\mathbf{x}) = 1 + \frac{9}{m-1}\sum_{i=2}^{m} x_i \,, \; h(\mathbf{x}) \; = 1 - \left(\frac{f_1(\mathbf{x})}{r(\mathbf{x})}\right)^2 \tag{4}$$

Optimization results are shown using computed trade off plots in Figure 11. AEMOO is able to find a very well diversified and accurate front before the 10th iteration itself (470 samples). On the contrary, with a comparable population size of 48, direct NSGAII failed to obtain the Pareto front even after 50 generations. The population size for NSGAII had to be increased to find the actual front. PDR was able to locate the Pareto front with a sample size of 47 per iteration, but was slower than AEMOO, as it took 30 iterations (1410 samples) to obtain a front of comparable (but slightly worse) accuracy.

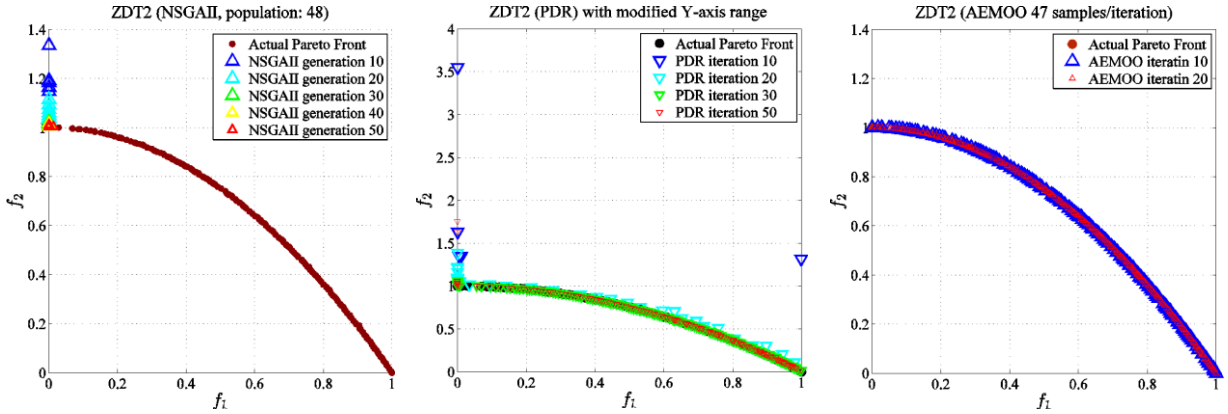

Figure 11: Example 3. Direct NSGAII (left), PDR (center) and metamodel-assisted AEMOO (right).

5.5 Example 5. Analytical example ZDT3 with 30 variables (Metamodel-assisted AEMOO)

This optimization problem is similar to Example 2, but has 30 variables instead of 10. The fronts using the three methods are shown in Figure 12, along with the actual one. AEMOO located all five disjoint regions within first 10 iterations, following which it enhanced the diversity. Both NSGAII and PDR were significantly slower and lacked accuracy. Using population size of 48, direct NSGAII completely missed 2 out of 5 regions. PDR was able to sample 4 of the regions satisfactorily at 40-50 iterations (1880-2350 samples). It found one non-dominated sample close to the fifth region, but not on it.

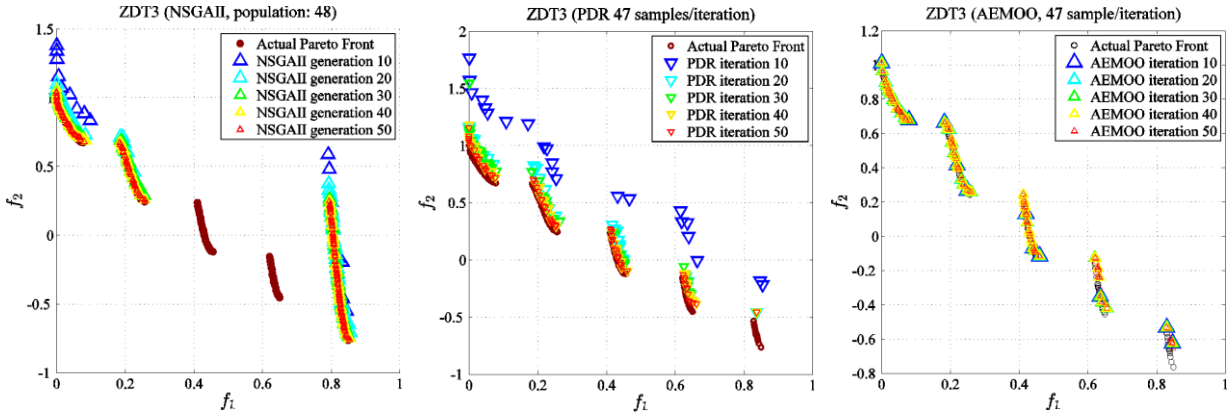

Figure 12: Example 4. Direct NSGAII (left), PDR (center) and metamodel-assisted AEMOO (right).

5.6 7 variable tolerance optimization of Chevrolet C2500 truck (Metamodel-assisted AEMOO)

AEMOO is used for tolerance-based MOO of a truck. Global metamodels are built using LS-OPT based on the truck's responses at 1500 samples (LS-DYNA). MOO is run on the metamodels. Relative tolerance and 6 thicknesses *x* are the variables. Mass is minimized and tolerance is maximized:

$$\begin{aligned} &\min_{tolerance,\bar{\mathbf{x}}} && \{-tolerance, scaled\_mass(\bar{\mathbf{x}})\} \\ &s.t. && P(scaled\_mass(\mathbf{x}) > 0.9) \le P_{\text{target}} \\ &&& P(scaled\_stage1\_pulse(\mathbf{x}) > 1) \le P_{\text{target}} \\ &&& P(scaled\_stage2\_pulse(\mathbf{x}) > 1) \le P_{\text{target}} \\ &&& P(scaled\_disp(\mathbf{x}) > 1) \le P_{\text{target}} \qquad \text{where } P_{\text{target}} \approx 0 \text{ in this work} \end{aligned} \tag{5}$$

In Figure 13, the vehicle parts to be optimized are shown along with the optimization results. The Pareto front obtained using AEMOO and NSGAII are shown. 100 samples per iteration are used to solve this example to ensure at least one sample of each class in the initial sampling. Other approaches to avoid this restriction are possible, but are outside the scope of this paper. AEMOO results are provided for 30 iterations that were completed at the time of writing this paper and compared to NSGAII is run up to 50 generations. The Pareto-optimal front using AEMOO has a better spread, and diversity compared to the NSGAII front even at the 50th generation, which shows its superior performance. At the same stage (30th iteration), the Pareto-optimal front using AEMOO is clearly better. The Pareto-optimal front consists of a knee at approximately 6% tolerance suggesting it to be a good design, as there is rapid mass increase beyond it.

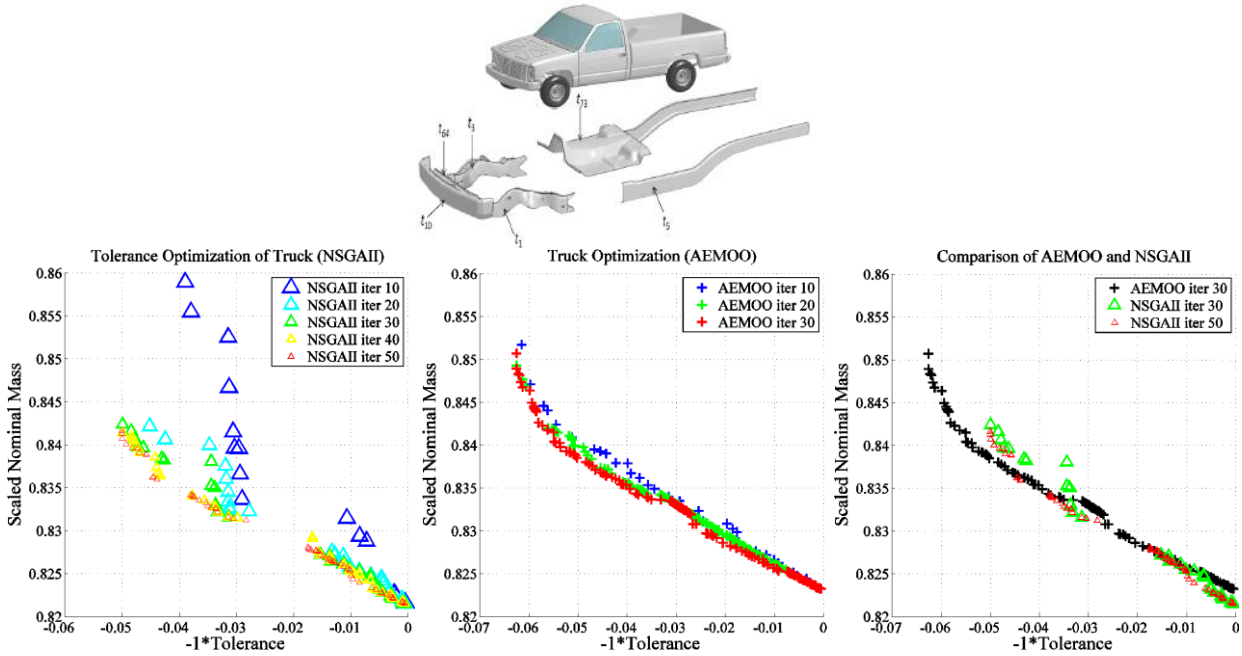

Figure 13: Truck to be optimized (top), NSGAII (bottom left), metamodel-assisted AEMOO (bottom center), and overlaid NSGAII and AEMOO fronts (bottom right).

6 Closure

An alternate or complementary approach to metamodel-based methods, which are used predominantly in the design community, is presented in this paper. This approach, based on classification methods, has been shown to possess several qualities that make it useful in certain types of problems. The advantage of this method in problems with discontinuous and binary responses was shown with the help of a binary problem. Additionally, particular emphasis has been given to a new classification-based MOO method, which has been shown to perform very efficiently. The proposed AEMOO method has several advantages compared to existing MOO methods due to its radically different approach. Two variations of the method are presented: direct and metamodel-assisted. The method's efficacy is validated using standard examples of up to 30 variables. It has been shown to clearly outperform existing methods like NSGAII and PDR in terms of the efficiency. Additionally, ability to locate disjoint Pareto fronts has been shown. Ability to solve constrained MOO has also been shown using a tolerance-based crashworthiness optimization example. As AEMOO explicitly defines the non-dominated region boundaries in the design space, it also naturally facilitates real-time Pareto optimality decisions. As the sampling scheme is partly based on design space classification, which discards significant regions of the space as dominated, it is expected to be affected by objective space dimensionality to a lesser extent. Future work is needed to validate this hypothesis. Additionally there is scope for further improvement in the constraint handling in AEMOO. The classification-based methods are currently being implemented in LS-OPT to provide a relatively seamless interface instead of using a third-party software. As a first step, the method is being implemented to perform reliability assessment by defining a new entity type "classifier" and by implementing an SVM classification algorithm. This will be followed by its enhancement to represent constraints in optimization as well as the implementation of AEMOO.

7 Literature

[1] Wang, G. Gary, and S. Shan. "Review of Metamodeling Techniques for Product Design with Computation-intensive Processes." *Proceedings of the Canadian Engineering Education Association* (2011).

[2] Stander, N., Roux, W.J., Basudhar, A., Eggleston, T., Craig, K.J. *LS-OPT User's Manual*, June 2014.

[3] Jones, Donald R., Matthias Schonlau, and William J. Welch. "Efficient global optimization of expensive black-box functions." *Journal of Global optimization*13.4 (1998): 455-492.

[4] Knowles, J. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. Technical report TR-COMPSYSBIO-2004-01, September 2004.

[5] Bichon, Barron J., et al. "Efficient global reliability analysis for nonlinear implicit performance functions." *AIAA journal* 46.10 (2008): 2459-2468.

[6] Youn, Byeng D., and Pingfeng Wang. "Bayesian reliability-based design optimization using eigenvector dimension reduction (EDR) method." *Structural and Multidisciplinary Optimization* 36.2 (2008): 107-123.

[7] Basudhar, Anirban, Samy Missoum, and Antonio Harrison Sanchez. "Limit state function identification using support vector machines for discontinuous responses and disjoint failure domains." *Probabilistic Engineering Mechanics*23.1 (2008): 1-11.

[8] Lee, Herbert KH, et al. "Optimization subject to hidden constraints via statistical emulation." *Pacific Journal of Optimization* 7.3 (2011): 467-478.

[9] Missoum, Samy, Palaniappan Ramu, and Raphael T. Haftka. "A convex hull approach for the reliability-based design optimization of nonlinear transient dynamic problems." *Computer methods in applied mechanics and engineering*196.29 (2007): 2895-2906.

[10] Basudhar, Anirban, et al. "Constrained efficient global optimization with support vector machines." *Structural and Multidisciplinary Optimization* 46.2 (2012): 201-221.

[11] Drucker, Harris, S. Wu, and Vladimir N. Vapnik. "Support vector machines for spam categorization." *Neural Networks, IEEE Transactions on* 10.5 (1999): 1048-1054.

[12] Jain, Anil K., Robert P. W. Duin, and Jianchang Mao. "Statistical pattern recognition: A review." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22.1 (2000): 4-37.

[13] Schuldt, Christian, Ivan Laptev, and Barbara Caputo. "Recognizing human actions: a local SVM approach." *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. Vol. 3. IEEE, 2004.

[14] Vapnik, V.N., and Vapnik, V. *Statistical learning theory*. Vol. 1. New York: Wiley, 1998.

[15] Hurtado, Jorge E. "Filtered importance sampling with support vector margin: a powerful method for structural reliability analysis." *Structural Safety* 29.1 (2007): 2-15.

[16] Basudhar, Anirban, and Samy Missoum. "Adaptive explicit decision functions for probabilistic design and optimization using support vector machines."*Computers & Structures* 86.19 (2008): 1904-1917.

[17] Basudhar, Anirban, and Samy Missoum. "An improved adaptive sampling scheme for the construction of explicit boundaries." *Structural and Multidisciplinary Optimization* 42.4 (2010): 517-529.

[18] Bourinet, J-M., F. Deheeger, and M. Lemaire. "Assessing small failure probabilities by combined subset simulation and support vector machines."*Structural Safety* 33.6 (2011): 343-353.

[19] Basudhar, A. Multi-objective Optimization Using Adaptive Explicit Non-Dominated Region Sampling. *11th World Congress on Structural and Multidisciplinary Optimization*, Sydney, Australia, 2015.

[20] Deb, Kalyanmoy, et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II." *Evolutionary Computation, IEEE Transactions on* 6.2 (2002): 182-197.

[21] Stander, N. An Adaptive Surrogate-Assisted Strategy for Multi-Objective Optimization. *10th World Congress on Structural and Multidisciplinary Optimization*, Orlando, Florida, USA, 2013.

[22] Basudhar, Anirban, and Samy Missoum. "A sampling-based approach for probabilistic design with random fields." *Computer Methods in Applied Mechanics and Engineering* 198.47 (2009): 3647-3655.

[23] Canu, S., Grandvalet, Y., Gigue, V., and Rakotomamonjy, A. SVM and Kernel Methods Matlab Toolbox. INSA de Rouen, Rouen, France, 2005.

[24] www.ncac.gwu.edu/vml/models.html, National Crash Analysis Center. Finite element model archive, 2008.