Machine Learning based PVT Space Coverage and Worst Case Exploration In Analog and Mixed-Signal Design Verification

H. Lin, Z. Ye and A. M. Khan
Texas Instruments
12500 TI Blvd,
Dallas, TX 75243

***Abstract*-In this paper, a PVT coverage model based on support vector machines is proposed to serve as a metric for evaluating the reliability of PVT corners or configurations used in AMS design verification flows. Based on the coverage model, an adaptive and iterative method is proposed to explore worst cases in the PVT space by starting the learning process from simulation data obtained from the traditional PVT corner simulations. The proposed method is able to capture the actual worst cases in the PVT space with affordable simulation expenses as demonstrated in the examples.**

I. Introduction

Verifying a design under process-voltage-temperature (PVT) variations is an essential step in analog and mixed-signal (AMS) design verification. Traditionally, it is performed via simulating extreme cases, which are usually referred to as corners. For example, a common set of corners consists of combinations of maximum, typical and minimum values of temperature and voltage, respectively, and outermost process models represented as strong, weak, nominal and skew. Verifying designs via such corner combinations is based on the assumption or expectation that worst case scenarios are caused by extreme PVT conditions and can be captured by a set of corners. This is appealing in verifying pure digital designs since failures are mostly observable in those extreme cases and the cost is affordable due to the limited number of corners. However, in AMS designs, circuit behaviors are more complex considering the continuous and nonlinear nature of analog signals. Consequently, checking designs on limited number of corner simulations may not be sufficient in AMS verification.

On the other hand, simulating and checking AMS designs exhaustively in a continuous PVT space is neither scalable nor practical since the cost of simulations for complex AMS designs can be extremely expensive. This urges us to develop a new scheme to (1) come up with better corner combinations or PVT simulation configurations so that we can capture potential failures that are not discoverable in traditional corners, and (2) leverage the outcomes of those simulations in a more efficient way to provide reliability or coverage estimation for the conducted simulations in AMS verification.

In this paper, we develop an algorithm on top of the commonly used cross-corner method to achieve better PVT coverage. As shown in Fig. 1, the algorithm starts with collecting simulation data from the traditional extreme corner simulations. After the analysis of the data, it suggests new PVT configurations that are potentially useful in an adaptive and iterative manner under the active learning framework [1] adopted from the machine learning domain. New simulations will be conducted for the suggested configurations, and new data will be collected and

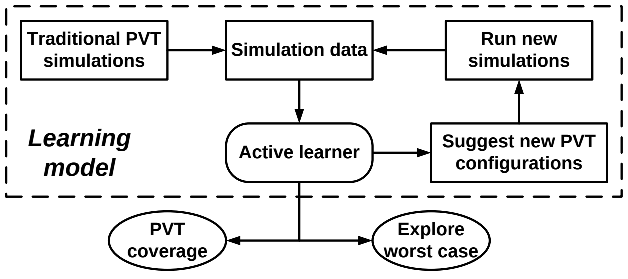


Figure 1. Overview of the proposed algorithm.

analyzed again. The process works like exploration of worst case or most risky PVT configuration for the given specification or performance requirements, and the final outcome of the algorithm provides PVT coverage as a metric to reflect the reliability of the simulation corners/configurations.

Compared to traditional methods, the proposed algorithm is able to approach closer to the actual worst-case scenarios of the AMS designs with an affordable overhead in terms of simulation need.

II. PVT Space Coverage

Since analog signals are continuous and usually nonlinear, the resulting circuit performance or specifications may be highly complex, and thus those commonly used extreme case corners (that is, combinatorial configurations of strong/weak process model, low/nominal/high voltage, and cold/nominal/hot temperature) may not always be sufficient to verify AMS designs. For example, in the AMS system shown in Fig. 2 (a), the specifications that need to be verified include the supply current, the output voltage of an LDO, and the output frequency of an oscillator. The three specifications measured at PVT grids with fine granularities are shown in Fig. 2 (b) (c) and (d), with different layers representing different extreme case process models. The supply current can be well verified using extreme case corners since it varies monotonically across the PVT space (Fig. 2 (b)). On the contrary, for the LDO output voltage and the oscillator output frequency, the actual peak values of both specifications in Fig. 2 (c) and (d) are higher than the maximal values obtained by those corners.

In fact, extreme case corners only check the design at the center (nominal) in addition to a few points on the boundary of the PVT space. This is not reliable since a point set with such a small size can hardly cover the whole continuous space, and the information provided by those corners is limited. It is an extremely challenging problem to ideally check 100% of the entire PVT space in the existing simulation based AMS design verification framework. Alternatively, it is more feasible to develop an estimation scheme of the PVT space coverage, and then find an

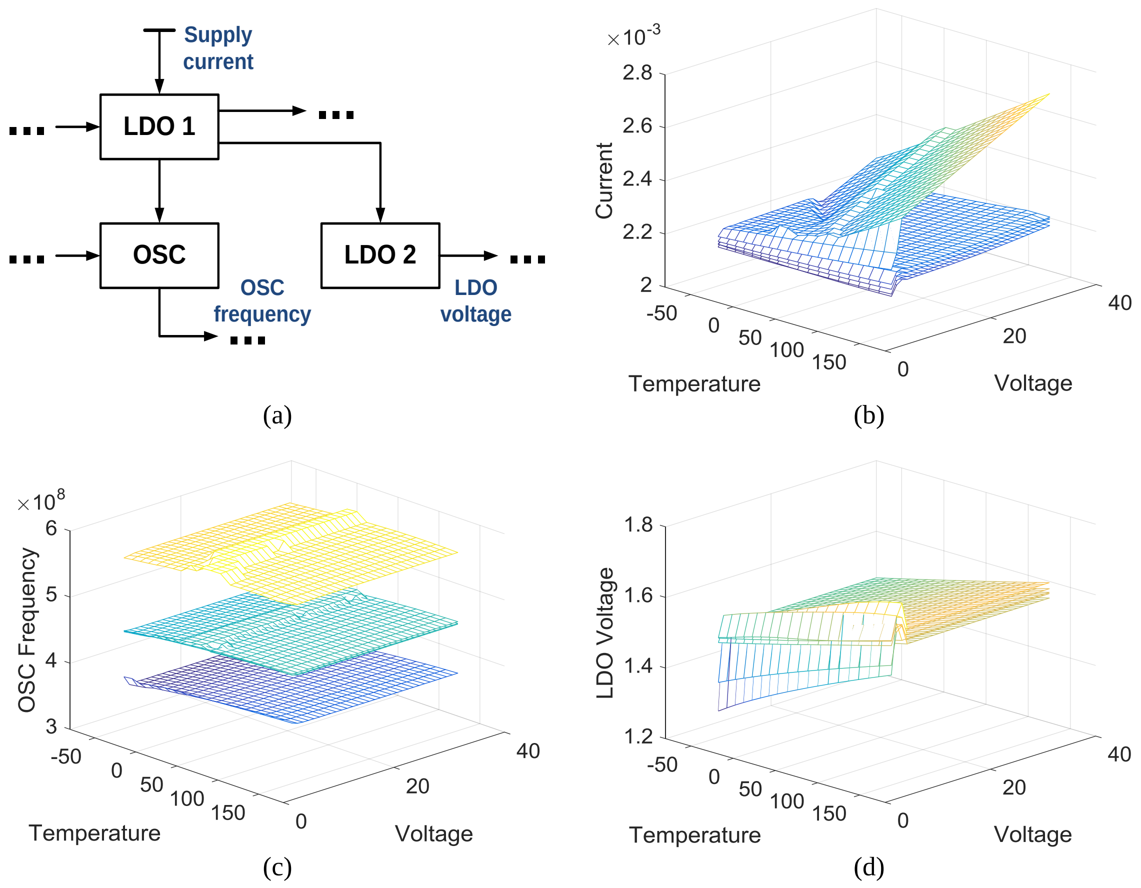


Figure 2. Example design and the simulation results in the PVT space: (a) example AMS design, (b) supply current, (c) oscillator frequency, (d) LDO voltage.

approach to achieve 100% PVT space coverage in an approximation sense. Such coverage estimation based methods are of great significance in improving the reliability of verification methodologies.

A straightforward approach towards PVT space coverage estimation and improvement is to increase the number of the simulations. For example, instead of using those extreme case corners, simulations can be conducted for a larger number of Monte Carlo random samples or fine-grained grids across the PVT space. However, an accurate estimation may require a huge volume of simulations, which may lead to unaffordable computational cost. To balance the coverage and the cost, we need to develop a more efficient method to leverage the information provided by the simulations.

Rather than viewing each simulation as an individual check of the design, collecting all the simulation data as a data set and performing mathematical analysis on the data set may offer us a key to the reliable and useful approximation of the mapping relationship, which links the PVT configuration of an arbitrary simulation to its resulting performance observation or specification measurement. By formulating such problems in a data driven context, there are powerful regression tools from the machine learning domain to solve them, such as the well-known support vector machines (SVMs) [2] and artificial neural networks (ANNs) [3].

Assuming that we have simulations conducted for PVT configurations (e.g. corners) $\{\boldsymbol{x}_i\}, i = 1, 2, \ldots, n$, and that for each configuration denoted by a vector $\boldsymbol{x}_i$, we have its corresponding design performance or specification measurement t_i, we can construct a mathematical function f, to approximate the actual mapping relationship $\cancel{f}: \boldsymbol{x} \rightarrow t$. Usually, PVT configuration will be bounded by constraints like upper and lower bounds of temperature and voltage, and limited number of extreme process corners, or process model with statistically bounded variabilities. We denote the set of all possible PVT configurations as Ω. For complex AMS designs, the mapping $\cancel{f}$ can be highly nonlinear across Ω.

An accurate f for the given data $\{\boldsymbol{x}_i\}$ will have small errors $\epsilon_i = |t_i - f(\boldsymbol{x}_i)|, i = 1, 2, \ldots, n$. And if the quality of $\{\boldsymbol{x}_i\}$ is good enough, we expect that the error $\epsilon = |\cancel{f}(\boldsymbol{x}) - f(\boldsymbol{x})|$ is also small for any $\boldsymbol{x} \in \Omega$. Given that the specification requirement is $t \in [t_{min}, t_{max}]$, based on the f derived from $\{\boldsymbol{x}_i\}$, we define the PVT space coverage estimation as:

$$PVT\ coverage = \frac{size\ of\ \{\boldsymbol{x} | t_{min} \le f(\boldsymbol{x}) \le t_{max}, \boldsymbol{x} \in \Omega\}}{size\ of\ \{\boldsymbol{x} | \boldsymbol{x} \in \Omega\}},$$

where $\{\boldsymbol{x} | t_{min} \le f(\boldsymbol{x}) \le t_{max}, \boldsymbol{x} \in \Omega\}$ denotes the set of PVT configurations whose simulation results are predicted to satisfy the given specification requirement. Its size ratio with the entire PVT configuration set can be used to evaluate the coverage of the approximation f which is produced by a limited number of PVT simulations. For example, a 100% PVT coverage means that the passing conclusion observed from the conducted PVT simulations can be extended to the entire continuous PVT space via the approximation f. On the other hand, a small PVT coverage implies that a notable portion of the PVT space may violate the given specification requirement based on the approximation f.

The computation of the size ratio of the two sets can be approximated by exhaustively collecting a large number of samples from Ω and evaluating them with f. Since f is usually a mathematical formula, the computational cost of evaluating $f(\boldsymbol{x})$ and approximating the coverage is usually negligible compared to simulation cost.

III. SVM Based Coverage Model

As illustrated in the previous section, the essential part in the PVT space coverage estimation model is the construction of the approximated mapping f. In this section, we develop the coverage model under the context of SVM, since SVM is a powerful machine learning toolbox with capability of handling high dimensional nonlinear problems, which makes it a good fit to the approximation of f. It should be noted that the approximation problem in the proposed coverage model is quite general, and we believe other nonlinear machine learning techniques such as ANN can also serve this purpose. The reasons for choosing SVM as our machine learning model are: (1) it is a classical learning algorithm with mature solvers available; (2) it is computationally efficient; (3) it is still one of the top machine learning algorithms in recent study [4].

For linear regression problem, SVM models f as a linear function $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} + b$, and the regression problem is formulated as a quadratic optimization problem [2], with $\boldsymbol{w}$ being the decision varialbes:

$$Minimize\ \frac{\|\boldsymbol{w}\|^2}{2} + C \sum_{i=1}^{n} (\xi_i + \xi_i^*),$$

$$subject\ to\ \begin{cases} t_i - \boldsymbol{w} \cdot \boldsymbol{x}_i - b \le \epsilon + \xi_i \\ \boldsymbol{w} \cdot \boldsymbol{x}_i + b - t_i \le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases}$$

where $\boldsymbol{x}_i$ is a training sample (a vector denoting the PVT configuration in our application) with target value t_i (the performance or specification measurement obtained from the simulation). The model parameter C is used to control the trade-off between the regularization term $\|w\|^2/2$ and the error term $\sum_{i=1}^{n}(\xi_i + \xi_i^*)$. The other parameter ϵ is used to determine the tolerance of the model. A well-selected set of C and ϵ can yield an accurate f and help get rid of the over-fitting problem.

The dual form of the above optimization problem using Lagrange multiplier α_i and α_i^* can be expressed as:

$$Minimize\ \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)(\boldsymbol{x}_i \cdot \boldsymbol{x}_j) + \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{n} t_i(\alpha_i - \alpha_i^*),$$

$$subject\ to \begin{cases} \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases}$$

and the resulting function f becomes

$$f = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\,(\boldsymbol{x}_i \cdot \boldsymbol{x}) + b.$$

The capability of solving nonlinear problems comes from the so-called kernel trick of SVM. Rather than solving the problem in the original space, SVM maps all the training samples into a higher dimensional space and then seeks a linear solution in that space. The projection of the higher dimensional solution back to the original space can be nonlinear. This mechanism is very flexible and the mapping can even be implicitly defined by a kernel function representing the inner product formula in the higher dimensional space, which is sufficient in solving the above dual problem.

A widely used kernel function is the RBF, also known as the Gaussian kernel:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) = e^{-\gamma\|x_i - x_j\|^2}.$$

The dimension of its implicitly mapped space is infinite, and there is only one tuning parameter γ in the kernel function, making it easy for model parameter selection. Typically, cross validation is employed to determine a set of proper model parameters including C, ϵ and γ.

By assembling assorted techniques together, the flow for the SVM based PVT space coverage estimation is shown in Fig. 3. We start from simulations for various PVT configurations and collect the measurements from the simulation results. Then, based on the collected data, we generate a training data set for SVM training. Note that quantities taken from AMS designs can vary in a large range (e.g. from Pico-scale to Giga-scale), data pre-processing such as normalization should be applied to the training data set to eliminate potential unintended biasing in the data.

Once the training data set is ready, cross validation can be employed to select proper values of the model parameters (i.e. the aforementioned C, ϵ, and γ if we use Gaussian kernel). With those selected parameters, SVM can be trained to obtain f as an approximation of the actual mapping between PVT configurations and their corresponding performances for the given AMS design. Leveraging the definition provided in the previous section, we can evaluate the PVT space coverage to reflect the reliability of the PVT configurations used in the data collection process.

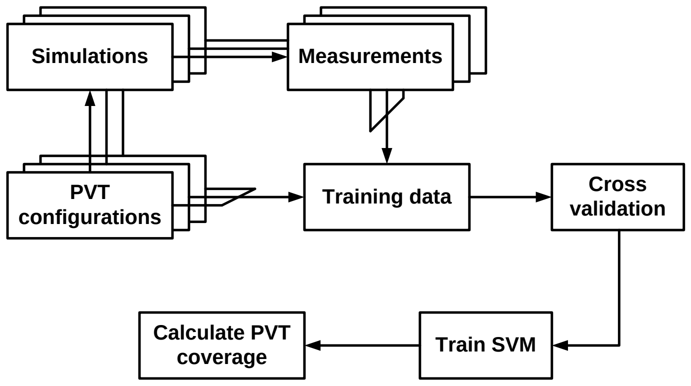


Figure 3. Calculation flow of the SVM based PVT space coverage.

Algorithm 1. SVM based PVT coverage evaluation

```
For each given PVT configuration $\boldsymbol{x}_i$:
    Run simulation, collect the resulting performance $t_i$;
    Add the pair $(\boldsymbol{x}_i, t_i)$ into the training data set $S$;
Normalize $S$;
For each given candidate set of model parameters:
    Perform 5-fold cross validation:
        Train an SVM with the candidate model parameters and 4/5 of $S$;
        Evaluate the error by testing the SVM model with the remaining 1/5 of $S$;
        Repeat 5 times;
    Record average testing error obtained from the 5-fold cross validation;
Select the set of model parameters with the smallest average testing error;
Train an SVM to obtain $f$ with the complete set $S$ and the selected model parameters;
Generate a large number of random possible PVT configurations $\hat{\boldsymbol{x}}_i$;
For each $\hat{\boldsymbol{x}}_i$:
    Calculate $f(\hat{\boldsymbol{x}}_i)$ as the performance prediction of $\hat{\boldsymbol{x}}_i$;
PVT coverage = ratio of $f(\hat{\boldsymbol{x}}_i)$ satisfying the specification requirement;
```

Our implementation uses LIBSVM [5], a widely used library for support vector machines, as our learning solver, and the pseudo code for the PVT coverage evaluation flow is as shown in Algorithm. 1.

IV. Coverage Improvement and Worst Case Exploration

Based on the definition, the quality of the estimated PVT space coverage relies on the quality of the approximation f trained from SVM, which is further dependent on the quality of the training data. An accurate model requires a notable amount of randomly selected training data that may still lead to expensive simulation cost for complex AMS designs. In this light, a smarter approach needs to be developed to reduce the need of simulations to make the flow more affordable.

Due to the regularization term in the optimization model of the SVM, it is likely that SVM will converge to a relatively sparse solution, meaning there are only a limited number of non-zero coefficients $(\alpha_i - \alpha_i^*)$ in the formula of f. Only those $\boldsymbol{x}_i$ with non-zero coefficients $(\alpha_i - \alpha_i^*)$ have contribution to the final construction of f and they are usually referred to as support vectors. This indicates that an SVM trained by a group of random samples may possibly be the same as, or very close to, another SVM trained by a small subset of those samples.

Moreover, specification requirements are usually given in the format of upper/lower bounds. For verification, we are more concerned about worst cases, that is, maximum/minimum values of the performance measured from the simulations, and whether those values comply with the pre-defined specification bounds. As a result, the approximation f should be accurate near the maximum/minimum values to suffice the need of verification, while it is allowed to be less accurate at locations far away from the maximum/minimum.

Taking advantage of these opportunities, we find potentials to develop more efficient ways to conduct our simulations. For example, for the artificial data shown in Fig. 4, where there is only one PVT parameter for better illustration, for the purpose of verification, there is no need to approximate the entire actual mapping (the blue curve) with high accuracy across the whole range of the PVT parameter. Instead, an approximation (like the red curve in Fig. 4) that gets highly accurate near the maximal and minimal performances, but gets less accurate at other locations, should suffice for verification of the defined upper and lower bounds of the specification. This suggests a potential sampling strategy (red stars in Fig. 4) that runs more simulations and collect more measurements near the maximum and minimum, but samples less intensively in the other regions of the PVT parameter space to reduce the simulation cost.

However, the maximum and minimum are usually unknown until we run the simulations in an intensive manner, which leads to the causality dilemma, or the chicken and egg situation: we need to run simulations to determine the maximum and minimum, while we need the maximum and minimum information to determine simulation configurations. To address the dilemma, we propose an iterative and adaptive method to sample new simulations and to explore worst cases for maximum and minimum simultaneously.

Starting with a small initial training data set, which can be generated by the traditional corner simulations, we train an SVM out of the data set. The resulting approximation may not be accurate, but it is the best bet of the mapping relationship in the context of SVM based on the data we have, and is capable of providing predictions of the maximum and minimum. We select the locations of the predicted maximum and minimum as the configurations

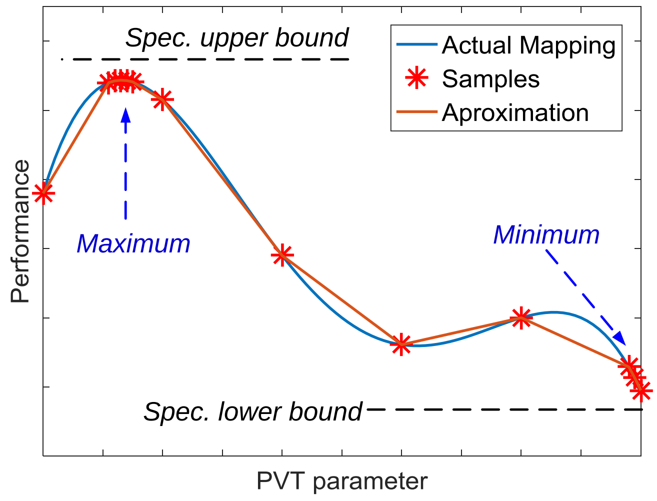


Figure 4. Potential reduction in number of required samples for verification.

to run new simulations, and then add the newly collected data into the training data set. The SVM model can be re-trained with this larger training data set, and the whole process can be repeated until the convergence occurs.

The intuition of the proposed method works as a repeating sequence of guess, check and learn. At early iterations, the guesses or the predicted worst cases may be not accurate due to the limited amount of training data. By adding data from actual simulations conducted for the guesses, the prediction model gets corrected and guided by the new data gradually. Once the outline of the prediction model roughly matches the actual mapping relationship, meaning the predicted worst cases may not be accurate, but are close to the actual worst cases, the new samples will improve the accuracy of the prediction model, and start aggregating and converging to the actual worst cases.

In addition, in the PVT coverage estimation proposed in the previous section, if the estimated coverage is unacceptable, there are basically two kinds of possible causes: (1) the quality of the approximation f is not good enough, or (2) there is indeed a defect/failure in the design so that certain PVT configurations will fail to comply with the specification requirements. In verification, we need to improve the quality of the approximation to enhance the coverage and the reliability for the first case, and find out potential bugs of the design for the second case. The proposed method can take care of both problems seamlessly. In the iterative process, the newly added simulations

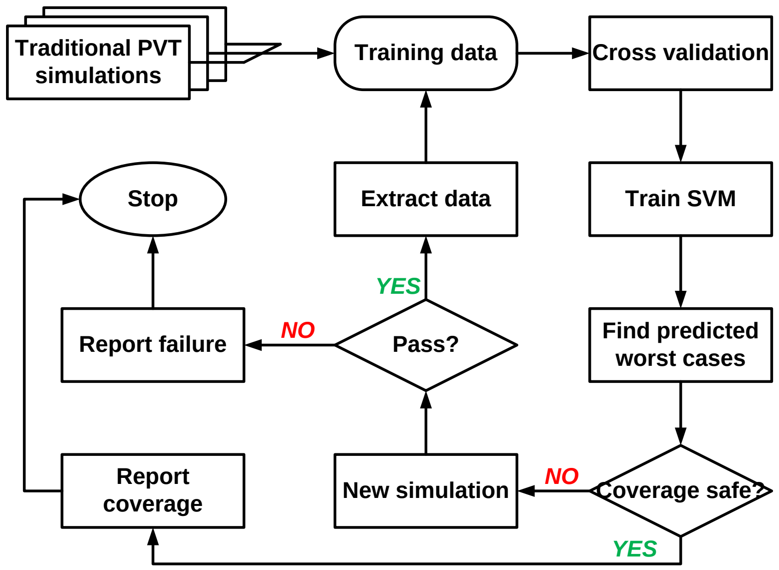


Figure 5. Coverage improvement and worst case exploration flow.

can help improve the quality of the approximation and push the predictions towards the actual worst cases. Therefore, as shown in Fig. 5, we develop a flow to improve the PVT coverage and explore worst cases simultaneously.

Initially, we use the traditional extreme corner PVT simulations to generate our training data set. Then, we select SVM model parameters via cross validation and train an SVM model with the data set. Using the learned prediction function, we can predict the locations of the worst cases in the PVT Space, as well as calculating the PVT coverage. If the coverage is safe (e.g. the prediction error is small and the coverage reaches 100% for a number of consecutive iterations), the iteration will stop and report the coverage. Otherwise, new simulation(s) for the predicted worst case(s) will be conducted and the simulation results will be checked based on the defined specification or performance requirements. If it fails to pass the checks, meaning this is a failure captured by the method, the process will stop to report the failure. If it passes the checks, new training data will be extracted from the simulation results and added to the training data set. The training, the prediction, the coverage calculation, and the simulation will be repeated until it either reaches a safe coverage or finds a failure.

V. Example

In this section, we apply the proposed flow to the system shown in Fig. 2 for verifying its oscillator frequency (Fig. 2(c)) and LDO voltage (Fig. 2(d)), where worst case cannot be captured by traditional PVT corners. For simplicity, we only define the upper bound for the two specifications, meaning the worst case exploration is the searching of the maximum value. The stopping criterion is whether all the predictions are lower than the existing

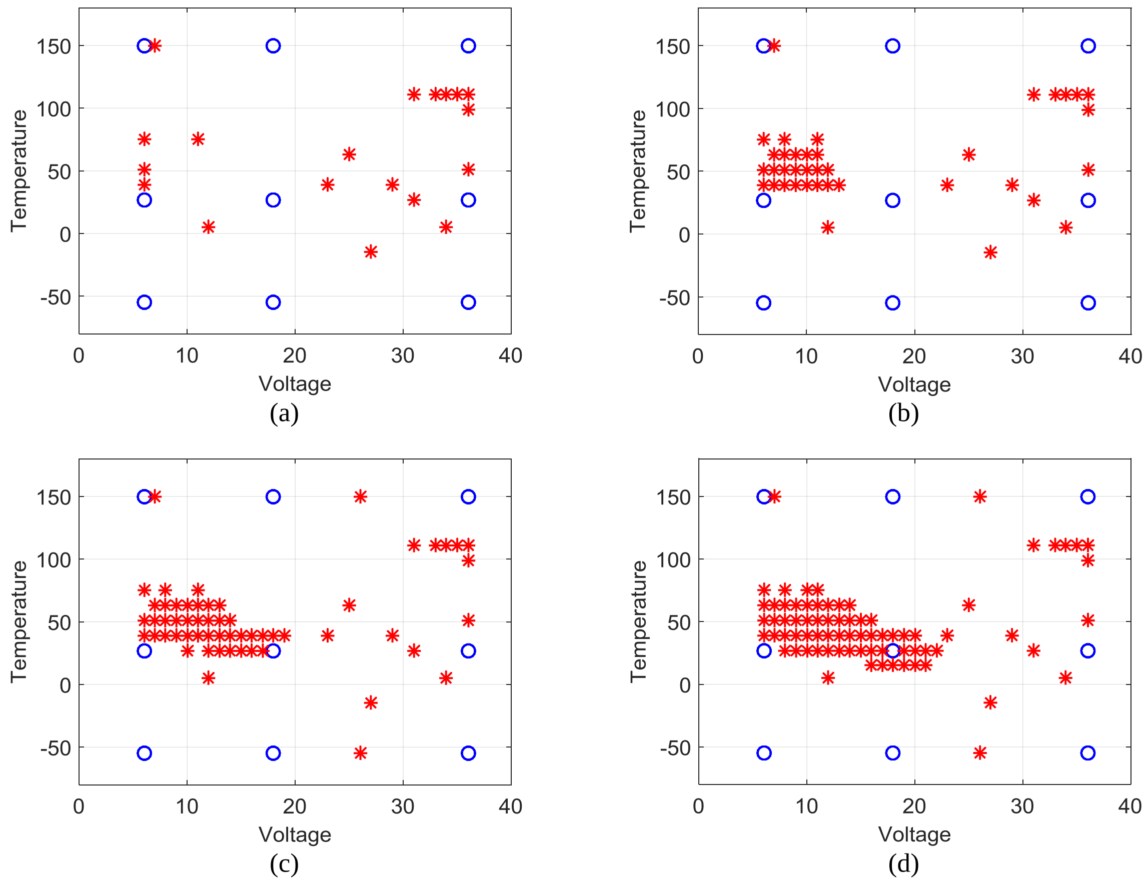


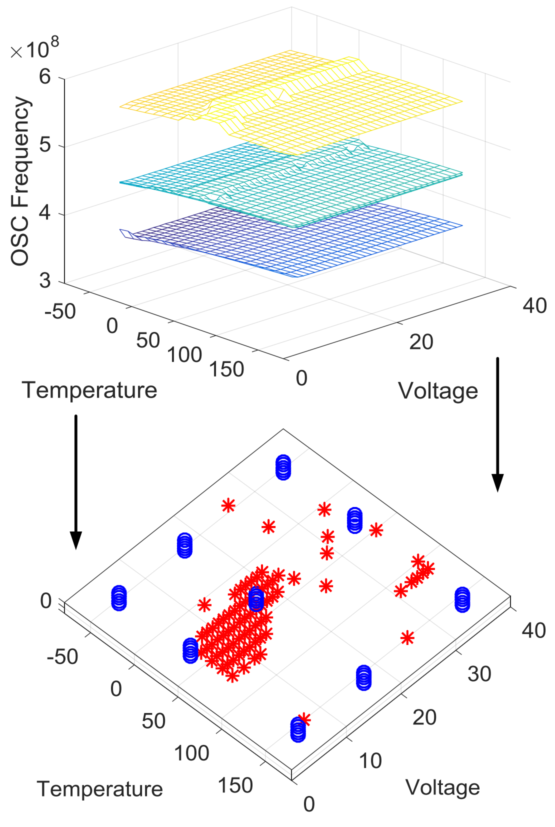
Figure 6. Adaptive sampling process for oscillator frequency, with blue circles representing the initial corners and red stars representing the new samples: (a) at the 19th iteration; (b) at the 38th iteration; (c) at the 57th iteration; (d) at the 76th iteration, this is the last iteration.

maximum value (i.e. 100% PVT coverage) for 20 consecutive iterations, or whether a failure is found based on the given specification requirements.

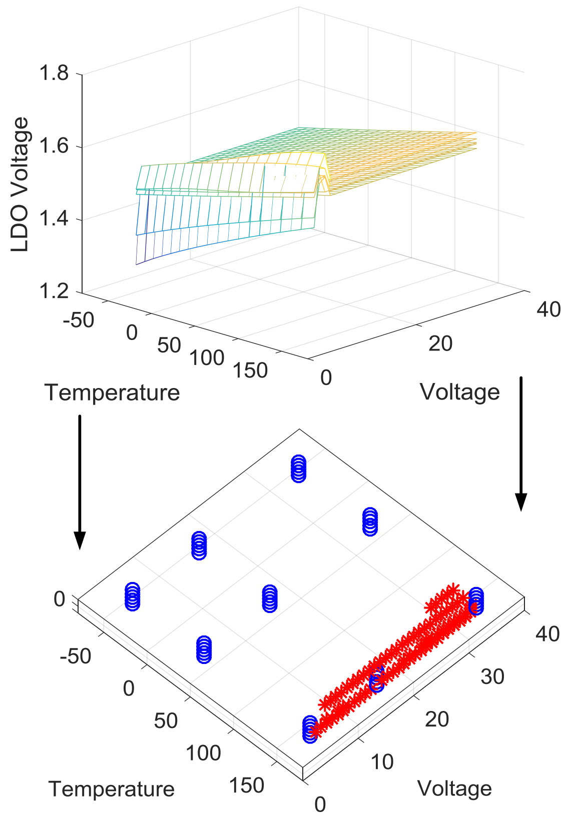
Fig. 6 shows the adaptive exploration process of the proposed method. All the simulation configurations are projected to a 2-dimensional space consisting of supply voltage and temperature. Blue circles denote the initial data used in the traditional extreme corners, including 5 process models, 3 voltage values and 3 temperature values, resulting in a total of 45 corners and 9 blue circles on the projected plane. Red stars denote the configurations suggested and then simulated by the method. As Fig. 6 demonstrated, at those early iterations, the SVM model is not accurate enough, and there is no obvious trend for the resulting suggested samples. However, the SVM model gradually gets more accurate and the method starts focusing the samples on the risky area, which is indeed where the oscillator frequency is higher as Fig. 7(a) reflected.

Similarly, Fig. 7(b) demonstrates that the method also shows intelligence in the sampling process when it is applied to the LDO output voltage. Most simulations are conducted at locations around the maximum. In both scenarios, the method is able to suggest samples closer and closer to the maximum, and finally capture the actual worst case in the PVT space.

The cost of locating the worst case via the proposed algorithm is much cheaper than checking all the uniformly selected grids. For example, assuming the uniformly selected corners have equal resolution in the temperature and the supply voltage, and there are a total of 5 extreme process models, to capture the maximum value in Fig. 1(c) and Fig. 1(d), it needs 605 simulations and 1280 simulations, respectively. Using the proposed algorithm, it only needs a total of 76 simulations and 139 simulations, respectively, including the 45 traditional corner simulations used in the initial training data set.



(a)



(b)

Figure 7. Correspondence of the performance and the samples selected by the proposed algorithm, with blue circles representing the initial corners and red stars representing the new samples: (a) for oscillator frequency, (b) for LDO voltage.

In this example, in addition to a smarter exploration strategy that yields higher efficiency by reducing simulation cost, the adopted learning mechanism also provides a reliable evaluation of the PVT coverage which is calculated via Algorithm 1. For both specifications verified above, the PVT coverage calculated via the SVM regression model in each iteration serves as an effective reliability evaluation, since it gradually increases as the flow keeps selecting more samples, and finally converges to 100% after the actual worst cases are reached.

VI. Conclusions

The method proposed in this paper is an assembly of existing machine learning techniques, and turns out to be a powerful tool when applied to the AMS verification flow. It provides estimation of PVT coverage as a metric of the reliability of the PVT corners/configurations, and is able to adaptively suggest and conduct new PVT simulations to explore actual worst case in the PVT space, which may not be covered by traditional corner simulations. As demonstrated in the two examples, the method accomplishes capturing the actual maximum with an affordable overhead in terms of number of simulations.

References

[1] H. Lin, and P. Li. "Circuit performance classification with active learning guided sampling for support vector machines," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34.9, pp. 1467-1480, 2015.

[2] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[3] C. M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.

[4] M. Fernández-Delgado, *et al*. "Do we need hundreds of classifiers to solve real world classification problems." *Journal of Machine Learning.* Res 15.1 (2014): 3133-3181.

[5] C. Chang and C. Lin, LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.