Automated Comparison of Analog Behavior in a UVM Environment

Sebastian Simon, Alexander W. Rath, Volkan Esen and Wolfgang Ecker
Infineon Technologies AG
85579 Neubiberg, Germany
{sebastian.simon, alexander.rath, volkan.esen, wolfgang.ecker}@infineon.com

***Abstract*—The comparison of analog behavior is still a time-consuming and error-prone procedure, which has do be done manually and therefore mainly relies on the expertise of the verification engineer. In order to accelerate and automate this process we introduce a new approach that enables us to quantify a degree of perceptual similarity between analog signals. This approach moreover follows the Universal Verification Methodology (UVM) standard and is integrated in a SystemVerilog library, which provides further functionalities for mixed-signal verification.**

***Keywords*—Mixed-signal verification, analog model validation, SystemVerilog, UVM, earth mover's distance**

I. Introduction

Due to better technology scaling of digital blocks compared to analog blocks more and more parts of the analog implementation of modern IC designs are shifted to the digital domain. However, certain analog components are still indispensable, leading to mixed-signal designs. In terms of verification, very few methodologies which consider the functional verification task of digital and analog blocks exist so far, whereas many verification methodologies for the digital domain have been developed; the newest being the UVM standard. Compared to the highly automated verification methodologies in the digital domain, verification in the analog domain implies a substantial amount of manual work and computational effort.

The high accuracy of SPICE-models is usually not essential for top-level verification. Hence, they are often replaced by appropriate behavioral models. This adjustment significantly speeds up the simulation. However, during the entire design process it has to be ensured that the behavioral model does not diverge from the respective analog component. This progressive model validation has to be done manually and is therefore a time-consuming and error-prone procedure.

In order to automate this validation we present a new approach to compare SPICE-models and behavioral models. For this purpose we are applying a metric which measures the distance between two feature vectors—in our case represented by UVM transactions. The algorithm is based on the earth mover's distance, which is widely used in content-based image retrieval. It computes the minimum costs for turning one transaction into another by solving a transportation problem, which in turn can be solved by linear optimization. This algorithm is implemented in SystemVerilog via DPI-C following the UVM standard. This measurement enables us to quantify a degree of similarity between an analog component and a reference model. Thus it is possible to automatically extract information about the behavioral match.

Furthermore, this approach is universally applicable such that it does not depend on a particular class of analog circuitry. In addition, the algorithm is independent of the transaction type. Hence, the transactions can be generated through sampling, Fourier analysis, or any other feature vector extraction method. This concept is moreover not limited to comparisons between behavioral models and SPICE-models, but can also be applied to any comparison of analog behavior such as two different SPICE-models or two behavioral models.

In order to demonstrate the aforementioned algorithm we show how it can be efficiently employed within a UVM environment. Also, we show how this environment is used to verify different behavioral models against their respective SPICE-models. Additionally, we present both passing and intentionally failing tests.

The paper is structured as follows. First, we give an overview of related work that covers approaches for comparing analog behavior. Their results form the basis for the motivation of our work. Following this, we briefly explain the components and functionalities of our analog UVM environment. In section IV we show the mathematical background of the earth mover's distance. In connection to that, the implementation of this algorithm is outlined. Finally we present the results of our approach in terms of accuracy and performance as well as the application to real-life models.

II. Motivation and Related Work

The employment of behavioral models as a substitute for analog models requires a comprehensive and accurate validation within an adequate test environment. Moreover any modification to the low-level model has to be detected as soon as possible, so that the behavioral model can be adjusted accordingly. Otherwise top-level simulations which include the corresponding high-level replacements would lead to wrong results and can have a determining influence on the overall effort of IC respins. If a design for example was simulated with a deficient behavioral model for a digital oscillator, all clocked blocks can be affected throughout the verification process. Such scenarios can be avoided by applying progressive regression tests and appropriate model validations.

If a SPICE-model shall be checked against its behavioral model the most common approach is to leverage assertions.

There are different works [1] [2] which successfully apply this technique for mixed-signal verification. However, assertions generally lack in abstraction since they have to be defined at signal level. Furthermore it is indispensable to reimplement them for the validation of new model types, so that they are highly application-specific. Therefore assertions should only be used as an ancillary technique for analog model validation.

An additional way to address this challenge is to make use of signal characteristics. The work of [3] introduces a flow for behavioral model validation which is based on user-specific checkers. In order to develop these checkers circuit characteristics and metrics have to be defined at first. Typical metrics can be the gain of an amplifier or output noise. However, this solution requires a recurring effort for the development of checkers since they have to be elaborated and implemented for each model validation.

Other works which apply both aforementioned approaches are presented in [4] and [5]. They propose a methodology called UVM-MS that includes extensions for, among others, analog signal checking and assertion techniques for analog properties. However, both methods show the same above stated deficiencies. They furthermore point out the challenge of automated analog comparison and the existing need to manually check waveforms.

All these approaches do not contain a universal technique for comparing analog signals. Considering for instance two different voltage regulator models which show the impulse responses depicted in figure 1. The first regulator (see top graph) is a Verilog-AMS model which is defined by the following transfer function:

$$G(s) = \frac{1}{1 + dTs + T^2 s^2}$$

where parameter d quantifies the damping and T denotes a time constant. The second graph shows the output of a real number model (RNM) implemented in SystemVerilog and based on corresponding difference equations.

Intuitively, both signals closely resemble each other, thus their curves can be classified as *similar*. Since this is only a subjective perception we want to introduce a formal metric which enables us to quantify such a characterization.

In addition further deficiencies of the aforementioned works shall be removed. On the one hand they compare outputs by evaluating particular signal characteristics which in turn means that assertions have to be defined at signal level. On the other hand there must be knowledge of signal thresholds, slopes, response time etc. so that checking components of the test bench depend on the type of circuitry. Hence we aimed at developing a transaction-based approach which is universally applicable to any type of circuitry.

III. Environment

Before dwelling on our approach, we introduce an extended UVM environment that we developed with the aim to verify analog behavior. We call this enhancement A-UVM (analog UVM). The main features of A-UVM are the usage of analog transactions [6], constrained-random analog stimulus [7] [6]

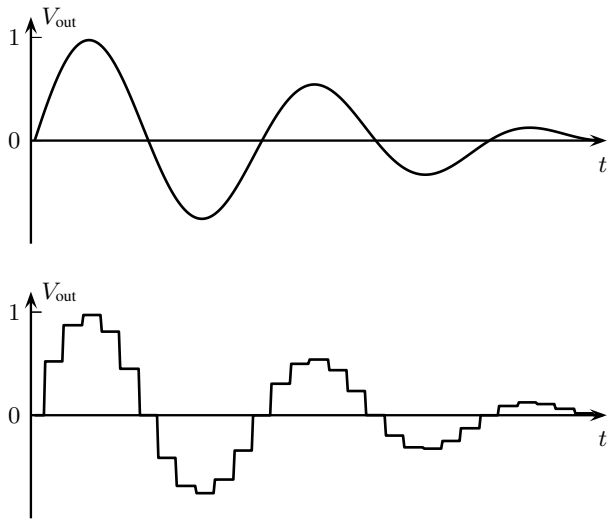

Fig. 1: Impulse response of two different voltage regulator models. The second model is less accurate since it produces a sampled output signal.

and the possibility of monitoring analog behavior [6]. In the following we briefly describe these key features.

A. Analog Transactions

Compared to the digital domain analog signals are particularly characterized by their unlimited co-domain. Hence, they can assume a specific shape as for instance linear, harmonic or even random. While digital signals can be easily transformed from bit level to transaction level due to their simple shape the abstraction of analog signals poses a major challenge.

However, in order to precisely describe an analog signal it is not sufficient to simply determine its shape. Thus additional parameters are necessary. For instance, in order to describe a linearly shaped signal, its slope as well as one value at a particular point in time have to be specified. Another example would be a sinusoidal signal which may be described by a number of real-valued parameters (e.g. amplitude, frequency, phase).

In A-UVM, we equate the term *shape* with the term *protocol* known from purely digital interfaces. Furthermore the term *transaction* represents a data structure which contains the required parameters to specify an analog signal.

B. Constrained-Random Analog Stimulus

In UVM-based environments drivers are used to translate transactions to digital signals which stimulate one or more DUTs. In this section we explain how A-UVM accomplishes this task for analog transactions defined in the previous section.

In order to transform analog transactions we implemented an interface for the communication between a generic driver and exchangeable *algorithms*. This interface allows new algorithms to be plugged in during the simulation. The algorithms in turn determine the type of transformation used by the driver.

They are not restricted to SystemVerilog, but can be written in C, Matlab or other languages as well.

Every analog transaction holds meta data which contains the algorithm name, among others. This information is read by the driver which subsequently selects the according algorithm from its data base. The transaction is then being passed to the algorithm in order to convert it to signal level.

C. Monitoring Analog Behavior

The previous definitions can also be used in terms of monitoring. An analog monitor has the task to extract signal parameters from a given shape. For this purpose the monitor uses an algorithm (later also referred as extraction method) via a dedicated interface.

Since the selection of a particular algorithm may be project-dependent we separated the monitor from the deployed monitoring algorithm. This enables us to manage all monitor functionalities in an own library. In addition to that, it allows one to exchange monitoring algorithms during run time. This may be useful in case signal shapes vary over time due to the DUT being in different states.

In order to enable the monitor to initiate the generation of transactions we added the concept of triggers to A-UVM. A trigger is an object that raises an event which is based on particular start activities like discontinuities, threshold crossings or simply a lapse of time. Hence, we implemented a selection of triggers covering frequently occurring activities. In addition, user-defined triggers can be plugged in via a callback mechanism. It is moreover possible to logically connect several start events and thereby create combinations of triggers. We finally leverage the same concept in order to terminate the generation of transactions.

IV. Mathematical Background

A. Bin-by-bin Measures

Various measures exist which can be used to define a distance between elements of different sets. The sets can be represented by distributions, histograms or vectors. When a degree of similarity between two signals has to be measured Pearson product-moment correlation or cosine similarity are often applied. Both can be categorized as bin-by-bin measures since they compare contents or corresponding elements, that is, they compare all x_i and y_j for $i = j$.

Another important bin-by-bin measure is the Tanimoto distance. Given two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ it is defined as follows [8]:

$$d_{\mathrm{T}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x}^{\intercal}\boldsymbol{y}}{\|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2 - \boldsymbol{x}^{\intercal}\boldsymbol{y}} \tag{1}$$

Unlike Pearson correlation and cosine similarity this distance can be usefully applied to one-dimensional vectors or scalars. In this case $d_{\mathrm{T}}(x, y)$ takes values in the range $[-0.\overline{3}, 1]$, hence it is bounded above and below. Because of these properties the Tanimoto distance will serve as so-called *ground distance* for our earth mover's distance in the following section.

However, all bin-by-bin measures have one crucial drawback. As mentioned previously, only pairs of elements which have the same index are matched, so that these measures do not incorporate information across bins. This in turn means, that they do not necessarily match perceptual similarity well [9]. This problem can be addressed by applying cross-bin measures like the earth mover's distance (EMD).

B. Earth Mover's Distance

The earth mover's distance is an approach to measure the distance between two multi-dimensional distributions. Informally, if one must successively transport soil from one pile to another in order to equalize them, the earth mover's distance calculates the minimum cost for the total transport. The distributions are also called *signatures* and represent weighted feature vectors of both output signals. The feature vectors in turn can be created by sampling, Fourier analysis or any other extraction method. In the following the concept of this approach, which is based on [10], shall be explained.

Given are two signatures X and Y where x_i, y_i denote elements from the respective feature vector. Each signature consists of n *clusters*.

$$X = \{(x_1, w_{x_1}), \ldots, (x_n, w_{x_n})\}$$
$$Y = \{(y_1, w_{y_1}), \ldots, (y_n, w_{y_n})\}$$

For each element x_i, y_i a particular weight w_{x_i}, w_{y_i} can be assigned. In our case a constant weight W is sufficient since no element shall be prioritized:

$$W = w_{x_i} = w_{y_i} = \frac{1}{n} \tag{2}$$

Moreover the possibility for transports between both signatures is defined as *flow* $\boldsymbol{F} = [f_{ij}]$, with f_{ij} the flow between x_i and y_j.

Now the idea is to find a flow $\boldsymbol{F}$ that minimizes the costs C for the overall work:

$$C(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{F}) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} \cdot f_{ij} \tag{3}$$

where d_{ij} denotes the ground distance between x_i and y_j. For our ground distance we chose the Tanimoto distance from equation 1 since this distance can be used for scalars and is bounded above and below. Aside from that it can be shown that a bounded ground distance results in a bounded earth mover's distance.

The optimization problem in equation 3 must furthermore satisfy the following constraints where constraint 5–7 can be simplified according to our assumption in equation 2:

$$f_{ij} \geq 0 \qquad 1 \leq i \leq n, 1 \leq j \leq n \tag{4}$$

$$\sum_{j=1}^{n} f_{ij} \leq w_{x_i} = \frac{1}{n} \qquad 1 \leq i \leq n \tag{5}$$

$$\sum_{i=1}^{n} f_{ij} \leq w_{y_j} = \frac{1}{n} \qquad 1 \leq j \leq n \tag{6}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{n} w_{x_i}, \sum_{j=1}^{n} w_{y_j}\right) = 1 \tag{7}$$

Constraint 4 allows only unidirectional flows from X to Y. Constraint 5 ensures that the weight in y matched to x_i does not exceed n^{-1}, while constraint 6 ensures that also the weight in x matched to y_j does not exceed n^{-1}. Finally, constraint 7 forces the total amount of weight matched to be equal to the overall weight of each signature.

In summary our metric consists in solving the following transportation problem for a pair of distributions on condition that the above stated constraints are satisfied:

$$d_{\text{EM}}(\boldsymbol{x}, \boldsymbol{y}) = \min_{\boldsymbol{F}=[f_{ij}]} C(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{F}) \tag{8}$$

This expression represents a linear optimization problem. Hence, it can be solved by an appropriate algorithm like the simplex method. The metric d_{EM} eventually yields a value within the interval $[0, 1]$ where 1 symbolizes complete dissimilarity and 0 a full match.

V. Implementation

Before leveraging the earth mover's distance the output signals of the models have to be transformed in such a way that two feature vectors of equal length are being extracted. The vectors must moreover contain enough information to obtain meaningful results.

Different methods can be chosen to perform this step. Already implemented types are sampling, Fourier analysis and the parameter extraction of standard signals like jump, ramp or sine. Other potential methods, which could be easily added, are for instance wavelet analysis or an extraction of mel-frequency cepstral coefficients (MFCC).

The actual feature extraction is done by generic monitors mentioned in section III-C. The selection of monitor algorithms mainly depends on the demands of the verification engineer and the requirements of the regarding project. In terms of similarity analysis, the employed algorithm defines the feature which is used for signal comparison within the scoreboard. The described concept is shown in figure 2.

In the following the realization of the most important classes and functions for the EMD implementation shall be explained.

First of all a SystemVerilog class for the earth mover's distance is defined (see listing 1). This class is used to instantiate the metric within the scoreboard. The inputs of the get-function are two feature vectors extracted from the model outputs. Moreover a direct programming interface (DPI) is defined in order to realize all CPU-intensive calculations as a C-implementation. For this reason the EMD class only contains the most essential parts. In addition to the feature vectors the C-function receives the size n of the vectors. A check ensures that both vectors have equal length.

```
import "DPI-C" function real
 a_uvm_lp_solve_dpi(bit [63:0]
 data_str_a[], bit [63:0]
 data_str_b[], int data_str_size);

class a_uvm_earth_movers_distance extends
 a_uvm_error_metric;

 `uvm_object_utils(
  a_uvm_earth_movers_distance)

 virtual function real get(
  a_uvm_data_str a, b);
       a_uvm_iterator_base a_iter,b_iter;
       int unsigned data_str_size;
       bit [63:0] data_str_a[];
       bit [63:0] data_str_b[];
       int i = 0;

       a_iter = a.get_iterator();
       b_iter = b.get_iterator();

       if(a_iter.get_size() !=
        b_iter.get_size())
       begin
         `uvm_fatal(get_type_name(),
          "The provided data structures
          must be of same size!")
       end

       data_str_size = a_iter.get_size();

       data_str_a = new[data_str_size];
       data_str_b = new[data_str_size];

       while(a_iter.has_next())
       begin
         data_str_a[i] = $realtobits(
          a_iter.next());
         data_str_b[i] = $realtobits(
          b_iter.next());
         i++;
       end

       return a_uvm_lp_solve_dpi(
        data_str_a, data_str_b,
        data_str_size);
 endfunction
endclass
```

Listing 1: EMD class implementation and DPI-C (in SystemVerilog)

Next step is to declare a separate C-function for the ground distance between elements x_i and y_i (see listing 2). This enables us to easily add new or exchange existing implementations for ground distances.

An important fact is that the earth mover's distance restricts the applied ground distance to become 0 in case x_i and y_i are equal. Hence, it is necessary to adjust the output interval for the Tanimoto distance. Since $d_{\text{T}}(x, y)$ attains its maximum for $x = y$ it has to be mapped from $[-0.\overline{3}, 1]$ to $[1, 0]$.

In order to solve the transportation problem stated in equation 8 we utilize an already existing solver for linear

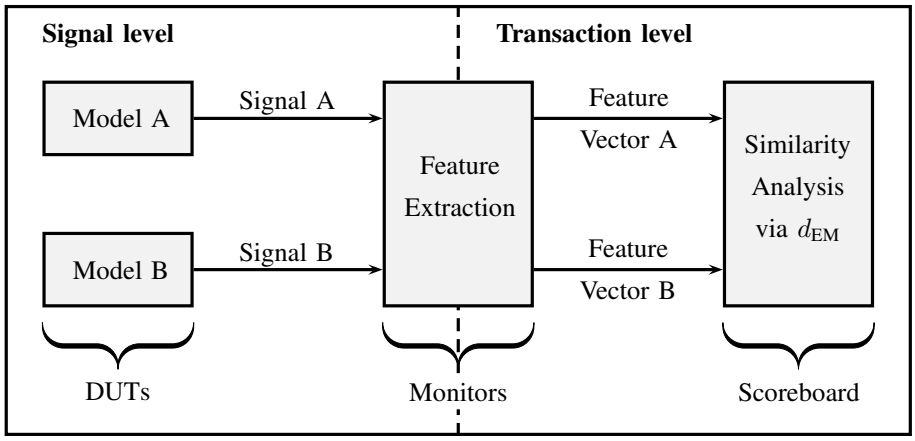

Fig. 2: Basic concept for feature extraction and similarity analysis

```
static double get_distance(double x,
    double y)
{
  if (x==0.0 && y==0.0)
    return 0.0;
  else
    return (-0.75*x*y)/(pow(x,2)+pow(y,2)
        -x*y)+0.75;
}
```

Listing 2: Mapped Tanimoto distance which is used as EMD ground distance (in C)

optimization problems, which is implemented in C [11]. It provides an API to define the objective function from equation 3 and the constraints 4-7. Furthermore its engine is based on the revised simplex algorithm, which is one of the most efficient algorithms in practice for solving linear optimization problems [12].

Listing 3 shows the main C-function for the earth mover's distance. The data structure *lprec* contains all information of the optimization problem and belongs to the solver API. First part of the function consists in calculating the ground distances across all elements which are further used to build the objective function.

```
double a_uvm_lp_solve_dpi(
 const svOpenArrayHandle data_str_a,
 const svOpenArrayHandle data_str_b,
 int data_str_size)
{
  int num_var=pow(data_str_size, 2);
  int i, j, row, col;
  double *aRow;
  lprec *lp;

  lp = make_lp(0, num_var);
  aRow=(double*) calloc(num_var+1,
   sizeof(double));

  // objective function
  for(i=0; i<data_str_size; i++)
  {
    for(j=0; j<data_str_size; j++)
        {
          aRow[i*data_str_size+(j+1)]=
          get_distance(
          *(double *)svGetArrElemPtr(
          data_str_a, i),
          *(double *)svGetArrElemPtr(
          data_str_b, j));
        }
  }
  set_obj_fn(lp, aRow);
}
```

Listing 3: EMD main function and calculation of ground distances (in C)

Finally constraint 5, 6 and 7 are implemented (see listing 4, 5 and 6, respectively). The remaining constraint 4 is already implicitly covered by the solver which only allows positive variables. With respect to inequation 5, 6 and equation 7 it can be said, that $2n+1$ constraints have to be defined for a particular optimization problem.

In the last step the solver optimizes the provided objective function under the given constraints. After another mapping from $[0,1]$ to $[1,0]$ the final result d_{mEM} is returned to the scoreboard and may take the following values:

- $d_{\mathrm{mEM}} = 1 - d_{\mathrm{EM}}(\boldsymbol{x}, \boldsymbol{y}) = 1$: Feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are identical.
- $d_{\mathrm{mEM}} = 1 - d_{\mathrm{EM}}(\boldsymbol{x}, \boldsymbol{y}) = 0$: Feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ behave completely contrary.
- $0 < d_{\mathrm{mEM}} = 1 - d_{\mathrm{EM}}(\boldsymbol{x}, \boldsymbol{y}) < 1$: Feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are neither identical nor behave completely contrary. However, the value for d_{mEM} quantifies their degree of similarity.

```
for(row=1, i=0; row<=data_str_size;
 row++, i++)
{
  for(col = 1; col <= data_str_size;
   col++)
        aRow[col+i*data_str_size]=1.0;
  add_constraint(lp, aRow, LE,
   1.0/data_str_size);
}
```

Listing 4: Implementation of EMD constraint 5 (in C)

```
for(row=1, i=0; row<=data_str_size;
 row++, i++)
{
  for(col=i+1; col<=pow(data_str_size, 2);
   col+=data_str_size)
        aRow[col]=1.0;
  add_constraint(lp, aRow, LE,
   1.0/data_str_size);
}
```

Listing 5: Implementation of EMD constraint 6 (in C)

```
for(col=1; col<=pow(data_str_size,2);
 col++)
  aRow[col] = 1.0;
add_constraint(lp, aRow, EQ, 1.0);
```

Listing 6: Implementation of EMD constraint 7 (in C)

VI. Application

When performing a similarity analysis for different models, feature vectors are extracted from their output signals and represented by transactions. Each pair of transactions is subsequently analyzed and assigned to a mapped EMD value d_{mEM}. Figure 3 shows an example of two analog signals where equidistant sampling is chosen as extraction method. The number of monitored samples per transaction is set to $n = 6$ (in the following also referred as number of data points). The extracted transactions are being compared pairwise in order to determine a distance d_{mEM} for each pair.

Now the idea is to define a lower bound for the expected similarity d_{mEM}. In case d_{mEM} falls below this bound the causing transactions can be located and examined in order to identify a potential behavioral mismatch between the employed models.

This test procedure can be run completely automated after setting up test cases and determining appropriate extraction methods. Furthermore the approach may be used to perform regression tests for the validation of analog models.

VII. Results

A. Performance and Accuracy

An application of our approach raises the question of performance and accuracy. Therefore we examined both aspects within a UVM test bench containing two analog models. Their input stimuli is derived from randomly generated transactions. Furthermore the extraction method is set to sampling.

For the performance analysis we measured the time consumed by the C-function of the EMD implementation. A considerable amount of time (approximately $90\,\%$) is thereby consumed by the solver itself in order to minimize the provided objective function. Figure 4 depicts the average execution time depending on the number of data points n used for the feature vectors. The graph shows a flat curve for data points in the range $[0, 134]$. However, it can be easily seen from the overall results that a nonlinear dependence exists between performance and n.

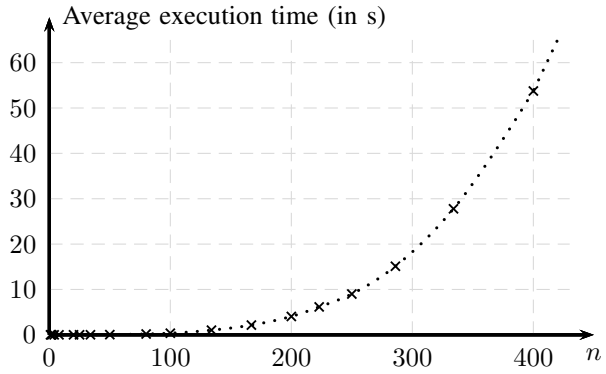

Fig. 4: Average execution time of the C-function depending on the number of data points (for one pair of transactions)

Considering equation 3 it can be moreover shown that the function has to be minimized for n^2 variables. This implies that a quadratic dependence exists between the length of feature vectors and the size of the objective function. As mentioned above the revised simplex algorithm is employed for minimization. However, this method shows an exponential worst-case but polynomial average complexity [13], which has a determining influence on the overall performance of our approach.

On the other hand, accuracy plays an important part when analyzing performance. For this purpose we determined the relative variation of the EMD results for each run. Figure 5 depicts the relative error depending on the number of data points n used for the feature vectors. As can be seen from the graph the curve is decreasing rapidly. Hence, accuracy can be significantly increased by using more data points.

For the purpose of a better comparison between execution times and relative errors, table I shows an overview for a selection of data points. In order to obtain meaningful results

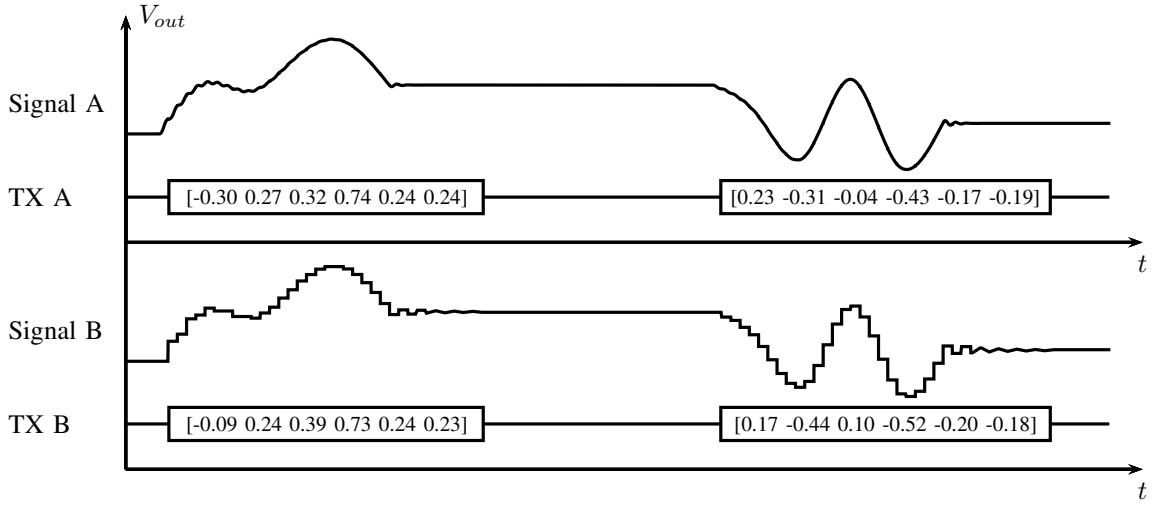

Fig. 3: Extracted transactions (TX) from two analog signals A and B by equidistant sampling

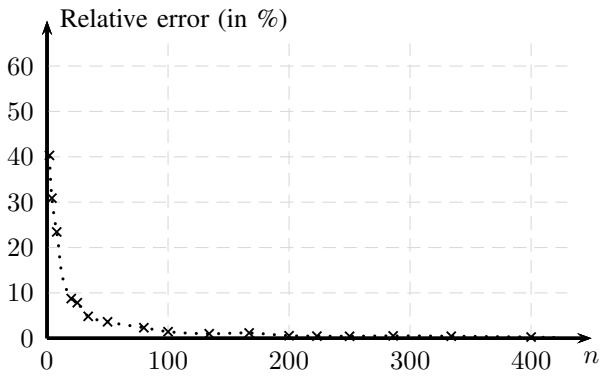

Fig. 5: Accuracy of d_{mEM} results depending on the number of data points

with a relative error below 1 % an execution time of circa 2 s has to be accepted which appears to be tolerable.

TABLE I: Performance and accuracy of the EMD approach

#Data points n	Average execution time	Relative error
20	0.0035 s	8.75 %
25	0.0055 s	7.83 %
34	0.0145 s	4.83 %
50	0.0430 s	3.63 %
80	0.1825 s	2.33 %
100	0.3590 s	1.44 %
134	1.0310 s	1.17 %
167	2.1500 s	1.00 %
200	4.0495 s	0.55 %

B. Application to Real Models

In order to demonstrate the successful application of our approach we would like to analyze two real-world models. For this purpose we refer to the models of the voltage regulators described in section II. As mentioned before both models show a similar output behavior. However, the second regulator model is less accurate due to its sampled output. In the following we quantify the degree of similarity between both of them and manipulate the output behavior of the second regulator model to force a lower similarity.

First of all we created four test cases which differ in sample time, output scaling and voltage offset. These values only apply to the second regulator model while the first model remains unmodified. Table II summarizes the test cases which were used for the similarity analysis.

TABLE II: Test cases for the validation of voltage regulators

Test case	Sample time	Scale factor	Offset
1	10 ns	1.0	0.0
2	40 ns	1.0	0.0
3	10 ns	3.0	0.0
4	10 ns	1.0	3.0

Both regulator models are being stimulated by random input signals, but with the same seed for each test case. Table III shows a selection of results for d_{mEM}. As expected, test case 1 yields the greatest values since no modification was done on both models. Increasing the sample time induces a slight but visible change in the results while scaling and offset show the highest impact on the similarity.

In terms of regression tests, it is conceivable to define an EMD lower bound as for instance $d_{\text{mEM}} > 0.95$. Hence, test case 2 would only flag a few pairs of transactions which would have to be reexamined, whereas test case 3 and 4 would reveal a serious behavioral mismatch between both models.

TABLE III: A selection of five calculated EMD values regarding the test cases in table II

d_{mEM} (case 1)	d_{mEM} (case 2)	d_{mEM} (case 3)	d_{mEM} (case 4)
⋮	⋮	⋮	⋮
0.9863	0.9693	0.6380	0.3993
0.9815	0.9649	0.6742	0.4675
0.9752	0.9635	0.6518	0.2524
0.9771	0.9305	0.6671	0.3283
0.9793	0.9622	0.6437	0.3305
⋮	⋮	⋮	⋮

VIII. Conclusion and Outlook

In this paper we presented a new approach for comparing analog models with respect to their output behavior. For this purpose we used a metric called earth mover's distance which was implemented in SystemVerilog and C following the UVM standard. The main idea is to quantify a degree of similarity between transactions which are extracted from analog signals. This measure enables us to perform regression tests in order to validate behavioral models by defining an EMD lower bound for all transactions. Hence, the effort for manually checking waveforms of analog signals can be reduced tremendously. Since this approach does not depend on the type of circuitry it is moreover universal applicable and highly reusable.

As a future work, we plan to employ our approach within further mixed-signal projects for the validation of analog behavior. In addition we intend to improve the performance of the solver engine, which is used to minimize our optimization problem. As a consequence the execution time would decrease while the accuracy of results would remain constant.

Our final goal is to provide a UVM-based model kit for the simulation and verification of analog designs. This model kit shall basically contain universal techniques for driving, monitoring and checking of analog signals as well as coverage collection and reference modeling.

References

[1] X. Yang, X. Niu, J. Fan, and C. Choi, "Mixed-signal System-on-a-Chip (SoC) verification based on SystemVerilog model," in *System Theory (SSST), 45th Southeastern Symposium on*, 2013, pp. 17–21.

[2] F. Neumann, M. Sathyamurthy, L. Kotynia, E. Hennig, and R. Sommer, "UVM-based verification of smart-sensor systems," in *Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), International Conference on*, 2012, pp. 21–24.

[3] V. Sharma, G. Lakshmanan, S. Tare, and S. Dhamankar, "Predicting the Correlation between Analog Behavioral Models and SPICE Circuits for robust SoC Verification," in *Behavioral Modeling and Simulation Workshop, Proceedings of the 2008 IEEE International*, 2008, pp. 130–135.

[4] N. Khan, Y. Kashai, and H. Fang, "Metric Driven Verification of Mixed-Signal Designs," in *Design and Verification Conference*, March 2011.

[5] N. Khan and Y. Kashai, "From Spec to Verification Closure: a case study of applying UVM-MS for first pass success to a complex Mixed-Signal SoC design," in *Design and Verification Conference*, March 2012.

[6] A. W. Rath, V. Esen, and W. Ecker, "A-UVM—A Transaction-Oriented UVM-Based Library for Verification of Analog Behavior," in *Design Automation Conference, 2014. ASP-DAC 2014. Asia and South Pacific*, January 2014.

[7] ——, "Analog Transaction Level Modeling for Verification of Mixed-Signal-Blocks," in *Design and Verification Conference*, March 2012.

[8] S. Theodoridis and K. Koutrumbas, *Pattern Recognition*, 4th ed. Academic Press, November 2008.

[9] Y. Rubner, C. Tomasi, and L. J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, November 2000.

[10] S. Cohen, "Finding Color and Shape Patterns in Images," Ph.D. dissertation, Standford University, May 1999.

[11] M. Berkelaar, K. Eikland, and P. Notebaert, "lp_solve," under GNU LGPL, May 2004, Open Source (Mixed-Integer) Linear Programming System.

[12] J. Matoušek and B. Gärtner, *Understanding and Using Linear Programming*. Springer, 2007.

[13] V. Klee and G. J. Minty, "How good is the simplex algorithm?" in *Inequalities–III: Proceedings of the Third Symposium on Inequalities Held at the University of California, Los Angeles, September 1–9, 1969*. Academic Press, 1972, pp. 159–175.