ABCD-L: Approximating Continuous Linear Systems Using Boolean Models

Aadithya V. Karthik[‡] and Jaijeet Roychowdhury
Department of Electrical Engineering and Computer Sciences, The University of California, Berkeley, CA, USA
[‡]Contact author. Email: aadithya@berkeley.edu

Abstract—**We present ABCD-L, a scalable technique for Analog/Mixed Signal (AMS) modelling/verification that captures the continuous dynamics of Linear Time-Invariant (LTI) systems, using purely Boolean approximations, to any desired level of accuracy. ABCD-L's models can be used in conjunction with existing techniques for Boolean synthesis/verification/fast logic simulation, or with hybrid systems frameworks, to represent LTI dynamics without incurring the penalty of adding continuous variables. Unlike existing state-enumeration approaches like DAE2FSM [1], ABCD-L scales practically linearly with system size. We apply ABCD-L to I/O links composed of RC/RLGC units, capturing important analog effects like inter-symbol interference, overshoot/undershoot, ringing, *etc.* – all using purely Boolean models. We also present a continuous-time differential equalizer example, where ABCD-L accurately reproduces key design-relevant AMS metrics, including the eye diagram correction achieved by the circuit. Furthermore, for real-world LTI systems, we demonstrate that ABCD-L can be applied in conjunction with Model Order Reduction (MOR) techniques; we use this to produce accurate Boolean models of an industry-scale power grid network (with 25849 nodes) made available by IBM. We also demonstrate that Boolean simulation using ABCD-L's models offers considerable speed-up over standard circuit simulation using linear multi-step numerical methods.**

I. Introduction

In today's advanced process technologies (32nm and below), Analog/Mixed-Signal (AMS) components (*e.g.*, interconnect, I/O and equalization circuitry, PLLs, DLLs, *etc.*) are becoming key bottlenecks that determine system-level performance [2], [3]. Moreover, an increasingly significant proportion of overall design bugs are now attributable to on-chip AMS components. For example, a recent internal study at Intel concluded that AMS modules account for over 20% of all design bugs in cutting edge microprocessors. Furthermore, such bugs tend to be difficult and costly to identify and correct, typically requiring extensive time-consuming SPICE-level simulations.

For early detection and timely correction of the above AMS-related design bugs, it is desirable to carry out functional validation and formal verification of AMS components *at or near SPICE-level accuracy*. However, in most existing approaches to AMS verification (see the accompanying supplement for an overview), the underlying model that is verified is usually a highly simplified abstraction that does not attempt to capture any of the SPICE-level subtleties (*e.g.*, layout-dependent parasitics, cross-talk, inter-symbol interference, ringing) that are responsible for bringing about design bugs/loss of performance. Thus, while the simplified models currently in use by AMS verification tools can be useful for gaining intuition about the circuit's operation as a whole (as intended by the designer), they are of limited use when it comes to *debugging* AMS designs or *issuing performance guarantees*.

Here, it is useful to draw a distinction between two kinds of formal verification techniques: *Boolean techniques* and *Hybrid systems techniques*. Boolean techniques (*e.g.*, [4]) represent each underlying circuit signal as a discrete (usually binary) quantity, whereas hybrid systems techniques (*e.g.*, [5]–[11]) offer the capability to represent signals as either discrete or continuous-valued quantities. However, the ability to represent and reason about continuous variables often comes at an enormous computational cost, which renders hybrid systems techniques typically orders of magnitude slower than their Boolean counterparts. Indeed, while Boolean techniques are routinely used in the industry to verify circuits with millions of logic gates, even state-of-the-art hybrid systems techniques are unable to verify systems with more than a few (*e.g.*, 5 to 10) continuous variables.

The *limited scalability* of hybrid systems techniques is the main reason why AMS circuits are typically not modelled/verified at SPICE-level accuracy; instead, existing hybrid systems methodologies are forced to adopt over-simplified "behavioural" AMS component models that often do not bear close resemblance to SPICE. This drastically limits their applicability in the context of AMS debugging/performance verification: without SPICE-accurate modelling, the predictions made by hybrid systems based AMS verification engines are not reliable enough for designers. As a result, the prevailing practice amongst AMS designers today is to carry out time-consuming SPICE simulations rather than place their trust in AMS verification tools. To overcome such "designer skepticism", we believe that it is necessary to significantly scale up existing hybrid systems techniques, so that they embrace SPICE-accurate models even for large AMS designs.

In this paper, we propose a technique (called ABCD-L[1]) to bridge the gap between SPICE-level detail and the models used by AMS verification engines, for an important subclass of AMS circuits, namely, Linear Time Invariant (LTI) systems[2]. The key idea behind ABCD-L is to approximate the continuous-time, continuous-valued dynamics of LTI systems using purely Boolean/discrete models. That is, given a set of differential equations for an LTI system (or alternatively, measured data, transfer function characteristics, scattering parameters, reduced order models, *etc.*), ABCD-L is a "push-button" style technique that produces as output a Boolean circuit abstraction (comprised entirely of Boolean logic elements such as registers, counters, *etc.*), that compactly encodes the analog behaviour of the given system, in a completely scalable fashion. Another important feature of ABCD-L is that it can approximate LTI systems to *any desired level of accuracy* (see §II).

Briefly, ABCD-L works by *discretizing* each underlying circuit signal, as well as the circuit's inputs, using as many bits as necessary to achieve the desired accuracy. These discretized values are stored in Boolean *registers*. Given the contents of each register at a particular time instant, ABCD-L uses Boolean logic to approximate the *next time instant* at which these contents must be updated (*e.g.*, in response to changing input). Thus, ABCD-L *transforms* the underlying LTI system into *an event-based discrete formulation*, realizable as a Boolean circuit (more details can be found in §II).

The above approach offers several compelling features. Firstly, because ABCD-L produces purely Boolean models, it is well-suited for use in conjunction with existing techniques for Boolean synthesis, verification, high speed logic simulation, *etc.* By reducing LTI dynamics to Boolean form, ABCD-L makes it possible to leverage powerful Boolean techniques for model checking/reachability analysis of LTI systems[3], as well as Boolean systems coupled with LTI dynamics (*e.g.*, high-speed digital logic with parasitic interconnect). Secondly, in the context of AMS modelling/verification, we know that existing hybrid systems approaches are unable to cope with more than a few continuous variables, whereas they can comfortably handle thousands of purely Boolean variables. Therefore, by enabling LTI dynamics to be accurately represented using "cheap" Boolean variables, ABCD-L frees up "precious" continuous variables for other purposes (*e.g.*, to accurately model non-linear dynamics). Therefore, with ABCD-L, it may be possible to expand the scope of hybrid systems approaches to much larger systems than they can handle at present. Although this notion has been theoretically studied before (*e.g.*, see [12]), we believe that ABCD-L constitutes the first practical approach for Booleanizing continuous LTI dynamics in an accurate, systematic, and scalable manner.

[1]**A**ccurate **B**ooleanization of **C**ontinuous **D**ynamics - **L**inear
[2]LTI systems constitute a fundamental class of AMS systems, including, for example, on-chip and off-chip interconnect, clock tree networks, filters, linearized small-signal circuits, channel models, *etc.*
[3]In this paper, we confine ourselves to the question of how to construct purely Boolean models that accurately capture analog LTI dynamics. *Formal verification* involving such models is an important next step, and one that is the subject of ongoing research. In this paper, however, we do not seek to address the verification problem.

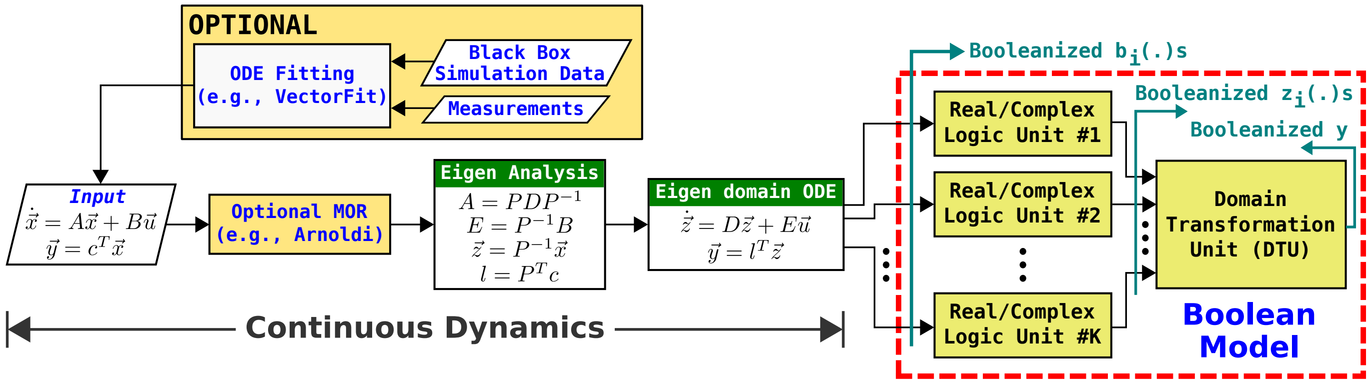

OPTIONAL
ODE Fitting (e.g., VectorFit)
Black Box Simulation Data
Measurements
Input
$\vec{x} = A\vec{x} + B\vec{u}$
$\vec{y} = c^T\vec{x}$
Optional MOR (e.g., Arnoldi)
Eigen Analysis
$A = PDP^{-1}$
$E = P^{-1}B$
$\vec{z} = P^{-1}\vec{x}$
$l = P^T c$
Eigen domain ODE
$\dot{\vec{z}} = D\vec{z} + E\vec{u}$
$\vec{y} = l^T\vec{z}$
Booleanized $b_i(.)$s
Booleanized $z_i(.)$s
Booleanized y
Real/Complex Logic Unit #1
Real/Complex Logic Unit #2
Real/Complex Logic Unit #K
Domain Transformation Unit (DTU)
Boolean Model
Continuous Dynamics

Fig. 1. The ABCD-L flow for producing a Boolean approximation that captures the analog dynamics of an LTI system. As shown in the figure, ABCD-L can also accept black-box simulation data as input, and it also integrates well with LTI MOR techniques.

Furthermore, ABCD-L also offers significant scalability advantages over explicit state-enumeration techniques like DAE2FSM [1], which produce Finite State Machines (FSMs) in State Transition Graph (STG) form, requiring exponential time and space complexity with respect to LTI system size. By contrast, ABCD-L's implicit, *circuit-based* models are exponentially more compact than STGs. As a result, ABCD-L scales practically linearly with respect to both system size and the desired fidelity of the Boolean approximation, without blowing up in either runtime or model size. In addition, the above event-based Boolean formulation lends itself to a new methodology for accurate, high-speed LTI system simulation (covered in §II). This methodology, as we show in §III, can offer considerable speed-up over traditional circuit simulation methods based on linear multi-step integration [13]. Moreover, ABCD-L has been designed to handle stiff systems efficiently, and it can take full advantage of LTI Model Order Reduction (MOR) techniques to Booleanize large LTI systems in a scalable manner.

We have applied ABCD-L to representative LTI circuits such as I/O links composed of RC and RLGC units. For these systems, we show that the Boolean models produced by ABCD-L are able to accurately reproduce the original system's continuous dynamics, including important performance-limiting analog effects such as inter-symbol interference, overshoot/undershoot, ringing, *etc.* In addition, we have applied ABCD-L to produce purely Boolean, small-signal models of a more complex, non-linear AMS system: an LTI channel followed by a differential equalizer (linearized using small-signal device models). In this case, too, ABCD-L is able to accurately capture and reproduce the circuit's dynamics in SPICE-level detail, using purely Boolean models all along. Further, ABCD-L is also able to capture higher-level design relevant AMS metrics, such as the eye diagram correction achieved by the equalizer. In addition, to accurately Booleanize industry-scale real-world LTI systems in a computationally viable manner, ABCD-L can be applied in conjunction with LTI MOR techniques; we demonstrate this for a 25849-node benchmark power grid network made available by IBM[4].

II. ABCD-L's Core: A New Technique for Booleanizing LTI Systems

In this section, we describe the key ideas behind ABCD-L.

Fig. 1 depicts the ABCD-L flow, which takes as input an LTI system, and produces as output a Boolean approximation for it. For simplicity, we assume that the LTI system is specified as an ODE[5], of the form:

$$\dot{\vec{x}} = A\vec{x} + B\vec{u}, \quad \vec{y} = c^T\vec{x}, \tag{1}$$

where $\vec{x}$ is the system's (continuous) analog state (a vector of voltages and currents), A is a real square matrix, $\vec{u}$ is the system's (time-varying) input, and $\vec{y}$ its corresponding output. We note that, if the ODE is not directly available, but instead only measured data (*e.g.*, from AC excitation at several frequencies), or S-parameters, or black-box transfer function characteristics, are available, ABCD-L can still obtain the requisite ODE by applying standard fitting techniques (*e.g.*, VectorFit [14], table-based methods [15], *etc.*).

As indicated in Fig. 1, ABCD-L begins with an eigenanalysis [16] of the above ODE system, which produces a new ODE system of the form:

$$\dot{\vec{z}} = D\vec{z} + E\vec{u}, \quad \vec{y} = l^T\vec{z}, \tag{2}$$

[4]Please see the supplement at the end of this paper.

[5]ABCD-L is also capable of Booleanizing more general systems of the form $Q\dot{\vec{x}} = A\vec{x} + B\vec{u}$, $\vec{y} = c^T\vec{x}$, where Q may or may not be invertible. However, due to space constraints, we limit our discussion here to LTI systems in ODE form.

where D is a square diagonal matrix containing the eigenvalues of A. The matrices $[A, B, c]$ are related to the matrices $[D, E, l]$ through the eigenvector matrix P (the relevant equations can be found in Fig. 1).

The i^{th} equation of the new ODE is a "de-coupled" scalar linear differential equation of the form:

$$\dot{z}_i = \lambda_i z_i + b_i(t), \tag{3}$$

where λ_i is the matrix entry $D_{i,i}$ and $b_i(.)$ is the i^{th} entry of the vector $E\vec{u}(.)$. Given the initial condition $z_i(t_0)$, the solution to the above equation is known analytically, and is given by:

$$z_i(t) = z_i(t_0)e^{\lambda_i(t-t_0)} + \int_{t_0}^{t} b_i(\tau)e^{\lambda_i(t-\tau)}d\tau \tag{4}$$

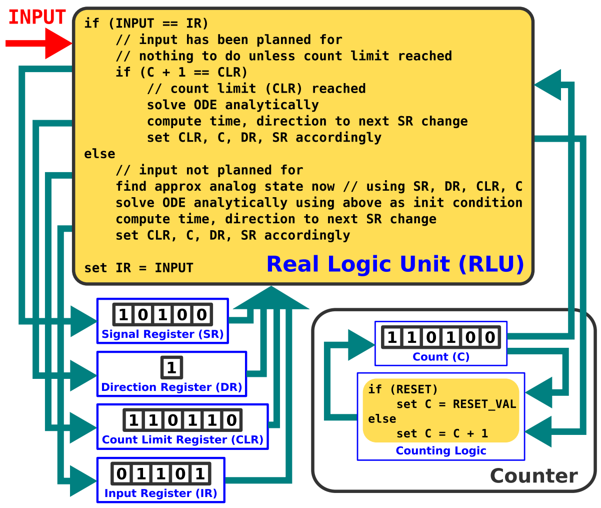
At this point, we would like to make some observations:

- On some (extremely rare, Lebesgue measure zero) occasions, the given LTI system may not be diagonalizable, *i.e.*, the matrix D above, instead of being diagonal, may take a Jordan form. It is possible (though tedious) to develop a general theory for Booleanizing such systems; however, because this almost never happens in practice, we do not consider it in this paper.
- In some cases, the size of the given system makes eigenanalysis impractical. In such situations, we first use an LTI MOR technique (*e.g.*, Arnoldi iteration [17], [18]), to reduce the system size, and then subject the reduced system to eigenanalysis. The theory of LTI MOR is well-developed, and many large LTI systems encountered in practice can successfully be reduced using the MOR techniques available today. Also, as Fig. 1 shows, it is relatively straightforward to integrate virtually any MOR technique into the ABCD-L flow; we demonstrate this for a 25849-node benchmark power grid network made available by IBM, in the supplemental material at the end of the paper.
- Many real-world LTI systems are stiff, *i.e.*, their underlying signals evolve at widely different timescales because the systems' eigenvalues span several orders of magnitude. ABCD-L can efficiently handle such systems by using different timescales to Booleanize the dynamics corresponding to different eigenvalues. However, due to space constraints, and for notational simplicity, we do not elaborate on this here.

Resuming the ABCD-L flow, the key idea behind ABCD-L is to *transform* the analytical solution of Eq. (4) into *Boolean operations* that can be expressed using digital logic constructs (registers, counters, *etc.*). This transformation is achieved by a set of Boolean *Logic Units (LUs)*, one for each component of $\vec{z}$. Each LU is either a *real LU (RLU)* or a *complex LU (CLU)*, depending on the corresponding eigenvalue. The output sequence of the i^{th} such LU is a (multi-bit) Boolean approximation of the i^{th} component of $\vec{z}(t)$. In addition, a combinational *Domain Transformation Unit (DTU)* combines the outputs of the LUs into a multi-bit Boolean approximation of $y(t)$ (which can be mapped back into a piecewise constant analog signal as a post-processing step). Below, we describe how the DTU and the individual LUs are structured.

The DTU's output is simply a Booleanized linear combination of its inputs. This is a combinational function whose Boolean specification/synthesis has been well-studied (*e.g.*, see [19]).

The LUs, on the other hand, are sequential systems, and their construction is more challenging. Each LU implements, using purely Boolean logic, a scalar linear differential equation $\dot{z} = \lambda z + b(t)$. The signals $z(t)$ and $b(t)$ are encoded as bit-vectors of length m (where m is a parameter passed to ABCD-L, called the *signal resolution*). The LU is designed so that its



```
if (INPUT == IR)
    // input has been planned for
    // nothing to do unless count limit reached
    if (C + 1 == CLR)
        // count limit (CLR) reached
        solve ODE analytically
        compute time, direction to next SR change
        set CLR, C, DR, SR accordingly
else
    // input not planned for
    find approx analog state now // using SR, DR, CLR, C
    solve ODE analytically using above as init condition
    compute time, direction to next SR change
    set CLR, C, DR, SR accordingly

set IR = INPUT
```

```
if (RESET)
    set C = RESET_VAL
else
    set C = C + 1
```

Fig. 2. Sequential logic schematic for discretizing a real scalar linear differential equation $\dot{z} = \lambda z + b(t)$ in the eigendomain.

bit-vector approximation to $z(t)$, when mapped back into the analog domain, closely matches the actual system response $z(t)$ for all input sequences $b(t)$. This is achieved by the logic structures depicted in Figs. 2 and 3, for real and complex eigenvalues respectively.

Let us first consider the real eigenvalue scenario, *i.e.*, the problem of Booleanizing $\dot{z} = \lambda z + b(t)$, where all quantities are real. Fig. 2 shows a Boolean schematic for this, which includes (a) a *Signal Register* SR that maintains an m-bit representation of z, (b) a 1-bit *Direction Register* DR, that denotes whether z is increasing/decreasing, (c) a *Count Limit Register* CLR that indicates the time at which SR must be incremented/decremented, (d) a set/reset counter with a count C, which measures time by counting up to the limit CLR, and (e) an m-bit *Input Register* IR that stores the input b. The whole unit is clocked at a pre-determined, fixed time-step Δ (in practice, it is usually straightforward to choose Δ to ensure that it is small enough to capture the dynamics of the scalar system above). For stiff systems, one can boost computational efficiency by using different Δs for the different LUs.

The above unit works as follows: as long as the input $b(t)$ remains constant, it is, in some sense, already "planned for". That is, the above structure stores enough information to know when, and in which direction, the register SR must be incremented/decremented. In this case, the count C keeps ticking up until it eventually becomes equal to the count limit CLR, at which time the register SR (and all the other registers as well) are updated accordingly. Analytical expressions, based on Eq. (4), are known for the "time to next change in SR mod Δ" operation, which can be either computed on the fly, or stored in a truth-table, *etc.*

For non-constant $b(t)$, as Fig. 2 indicates, the logic unit responds by considering the latest sampled input to be a new DC input, and updates all the registers using the analytical "time to next change mod Δ" function, making an intelligent estimate about the current value of z using the contents of registers SR, C, CLR, and DR (note that the count can be reset to any value by passing the RESET signal to the counter).

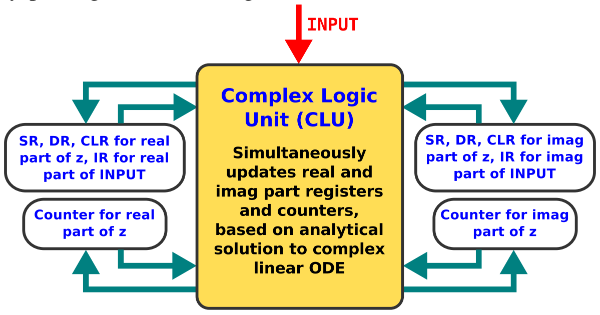

INPUT
Complex Logic Unit (CLU)
Simultaneously updates real and imag part registers and counters, based on analytical solution to complex linear ODE
SR, DR, CLR for real part of z, IR for real part of INPUT
Counter for real part of z
SR, DR, CLR for imag part of z, IR for imag part of INPUT
Counter for imag part of z

Fig. 3. Sequential logic implementation schematic for discretizing a complex scalar linear differential equation $\dot{z} = \lambda z + b(t)$ in the eigendomain.

For the complex eigenvalue case, a CLU (Fig. 3) essentially consists of two copies of each RLU register, storing the real and imaginary parts of each underlying signal. Whenever the registers need to be updated (for example, if the input has changed or if the limit CLR is reached by one of the counters), both the real and the imaginary sets of registers are updated simultaneously, based on the analytical solution given by Eq. (4).

From the above discussion, ABCD-L's accuracy clearly increases with the signal resolution m (which is a designer-specified parameter that determines how finely the underlying signals are discretized). In principle, this allows ABCD-L to abstract the given LTI system to any desired level of accuracy. In practice, for a designer, it is usually straightforward to determine an appropriate value for m through trial and error.

We also note that time-domain simulations involving ABCD-L's Boolean models can be very efficient, because they can be carried out entirely in the logical/Boolean domain (if necessary, using specialized logic simulation tools), without requiring any differential equation solving. To speed up simulation even further, we can devise an algorithm that jumps directly to the time instant specified by CLR, instead of incrementing the count C at every time-step. Indeed, we have implemented this algorithm, and in §III, we demonstrate that it can be significantly faster than conventional circuit simulation methods such as linear multi-step integration (even after accounting for the time taken up by eigenanalysis, model generation, *etc.*).

III. Results

Having described the core techniques behind ABCD-L, we now apply them to Booleanize LTI systems that are of interest to AMS designers, including, (1) I/O links modelled using RC/RLGC chains, and (2) an "LTI channel followed by a differential equalizer" circuit, linearized using small-signal analysis. We show that the Boolean models produced by ABCD-L are able to accurately capture the continuous-time dynamics of such systems, including important analog effects such as inter-symbol interference (ISI), overshoot/undershoot, ringing, *etc.* Also, we show that ABCD-L is able to faithfully reproduce higher-level AMS-design metrics for such systems, including the *entire shape of the eye diagram opening* at key circuit nodes. Furthermore, from a computational efficiency perspective, we show that ABCD-L can offer considerable simulation speed-ups over traditional LTI ODE/DAE simulation methods like linear multi-step integration (especially for larger LTI systems). Note that the examples in this section do not use LTI MOR; for examples involving MOR, please see the supplemental material.

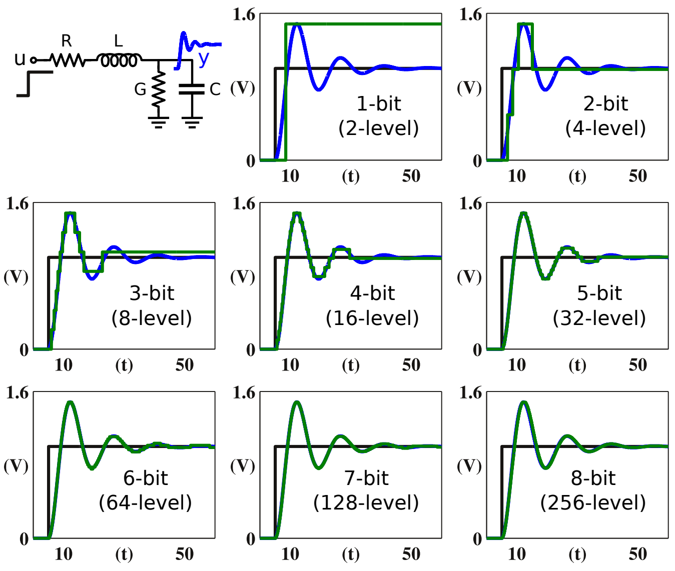


Fig. 4. ABCD-L applied to an RLGC filter, to produce Boolean models of arbitrarily high accuracy. The continuous system's response to a step input $u(t)$ (in black) is computed both analytically (blue), and using ABCD-L (green). In the plots above, the X-axis denotes time in RC units, and the Y-axis denotes voltages in Volts.

Before presenting the examples described above, we would first like to highlight an important feature offered by ABCD-L: in spite of using purely Boolean models, ABCD-L can reproduce the continuous-time dynamics of LTI systems with *arbitrarily high accuracy*, simply by increasing the number of bits used to represent the underlying circuit signals. Fig. 4 illustrates this using an RLGC filter (shown at the top left of the figure). This is a linear system of size 2, whose eigenvalues are both complex. As described in §II, ABCD-L discretizes the circuit's input $u(t)$ (in this example, a unit step function), its internal voltages/currents, and its output $y(t)$, into bit vectors of length m, where higher values of m correspond to finer quantization of the underlying analog signals (hence greater accuracy). The figure shows that, as we increase m from 1 to 8, the response predicted by ABCD-L's Boolean model (the green waveform) matches the actual system's response (the blue waveform) more and more accurately, duplicating important features such as overshoot and ringing. This is true for LTI systems in general: ABCD-L's Boolean approximations can be as close to the original system as desired.

We now apply ABCD-L to chains of RC/RLGC units (Fig. 5); these are often used to model high-speed I/O links, interconnect networks, on-chip communication channels, *etc.* [20], [21].

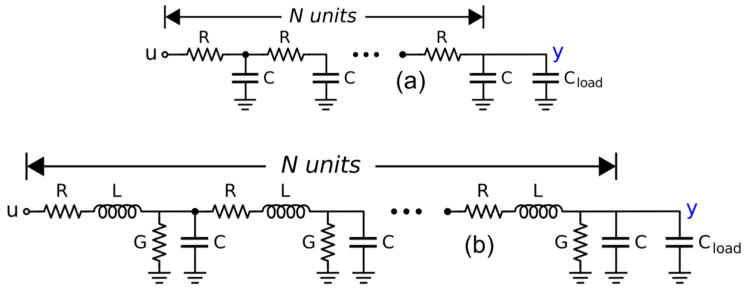


Fig. 5. RC and RLGC chains of length N, each driving a load capacitance C_{load}

Our experiments on RC/RLGC chains run as follows:

- We build an RC/RLGC chain, and apply several long, randomly generated bit patterns $u(t)$ at its input. To model both small and large inter-symbol interference (ISI), we vary the unit-interval T (the time that elapses between successive bits at the input), which is the inverse of the bitrate. Note that ISI decreases with increasing T, and vice-versa.
- We use ABCD-L to predict the system's responses $y(t)$ to the above inputs. Between experiments, we vary ABCD-L's signal resolution parameter m (the number of bits used by ABCD-L to quantize the underlying circuit's eigendomain signals).
- We compare ABCD-L's time-domain predictions against those of an ODE/DAE solver, plotting them on top of one another.
- Finally, since I/O link/interconnect engineering often makes extensive use of eye diagrams [22], we also convert ABCD-L's predictions into eye diagram form, and compare against eye diagrams generated by an ODE/DAE solver.

Fig. 6 depicts the results obtained by applying 4-bit ABCD-L to a 10-unit RC chain, and to a 10-unit RLGC chain, under conditions of both small ISI and large ISI. We note that, because all signals are quantized using 4 bits, the output of the system, as predicted by ABCD-L, consists of at most $2^4 = 16$ different levels. Moreover, as seen from the figure, in all cases, the 16-level predictions made by ABCD-L (drawn in green) closely match the system's actual continuous-time responses (drawn in blue). Therefore, the Booleanized models produced by ABCD-L appear to be good approximations of the underlying continuous LTI systems.

In parts (a) and (c) of Fig. 6, the bitrate of the applied input (the black waveform labelled $u(t)$) is low enough that the resulting ISI is small. This is readily seen from the system's responses: whenever the input bit is high (low), the output response has enough time to rise (fall) to a reasonably high (low) level before the next bit arrives. On the other hand, in parts (b) and (d) of the figure, we increased the input bitrates to a point where the induced ISI became significant. This can also be seen visually from the figure – even if an input bit is high (low), the output responses often do not have enough time to rise (fall) before the next bit arrives. As the figure shows, all these effects are captured quite accurately by ABCD-L.

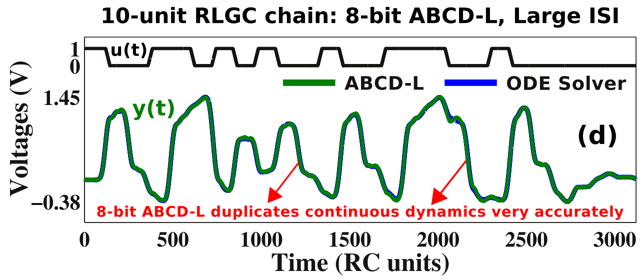


Fig. 7. Applying 8-bit ABCD-L to a 10-unit RLGC chain, for the same inputs as in Fig. 6(d). With the increased signal resolution, it is seen that the deviations between ABCD-L's prediction and the system's actual response are significantly reduced.

In Fig. 6(d), we have drawn attention (with a red circle) to a small time-interval where there is some deviation between ABCD-L's prediction and the system's response. Although such deviations tend to be "self-correcting" (as seen from the figure), it may be desirable to reduce the magnitude of these deviations. As described in §II, this can be achieved simply by increasing the number of bits used by ABCD-L to discretize the underlying waveforms (this corresponds to changing a single parameter that is passed as an argument to ABCD-L). This is illustrated in Fig. 7, where we have applied 8-bit ABCD-L (instead of 4-bit ABCD-L) to the same RLGC chain, for the same inputs as in Fig. 6 (d). From the figure, it is readily seen that the deviations between ABCD-L and the original system have been significantly reduced. This supports our earlier assertion that ABCD-L can model any LTI system with arbitrarily high accuracy.

Having illustrated ABCD-L's ability to closely match the underlying system's responses for short input patterns, we now simulate the ABCD-L-generated Boolean models on much longer input patterns (thousands of bits), and convert the resulting predictions into *eye diagram form*; this representation is extensively used in I/O link/interconnect design, modelling, analysis, and simulation, because it provides the design architect with a quick snapshot of key system properties [22]. Due to space constraints, we present eye diagrams only for the RLGC chain (and not the RC chain).

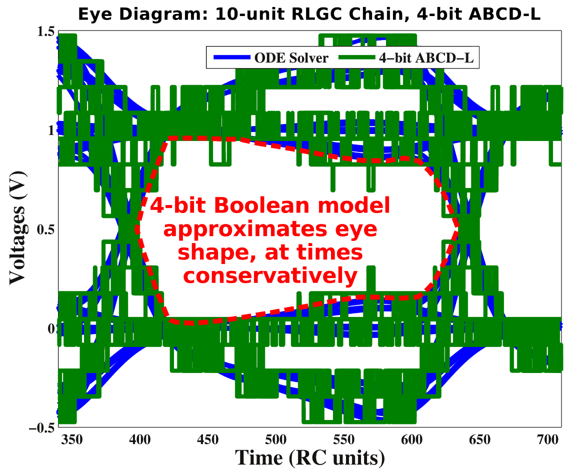


Fig. 8. Eye diagrams produced by 4-bit ABCD-L (green), and by an ODE solver (blue), for the 10-unit RLGC chain.

Fig. 8 depicts the eye diagram produced by applying 4-bit ABCD-L (in green) to the 10-unit RLGC chain of Fig. 6. This eye diagram is overlaid on top of the eye diagram produced by an ODE solver (which is in blue). The red dashed contour traces the shape of the eye opening, as predicted by the ODE solver. As seen from the figure, ABCD-L's 4-bit Boolean model is able to reproduce the entire shape of the eye opening with good accuracy.

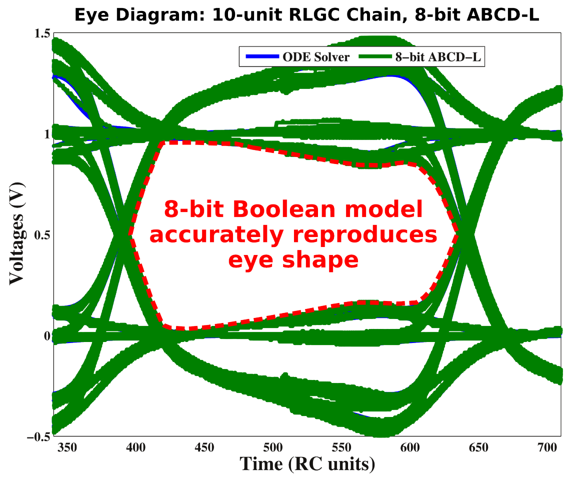


Fig. 9. Eye diagrams produced by 8-bit ABCD-L (green), and by an ODE solver (blue), for the 10-unit RLGC chain. It is seen that the shape of the eye opening is reproduced with increased accuracy compared to Fig. 8.

However, the eye opening produced by 4-bit ABCD-L is at times a bit conservative. While this may not be a problem for many applications, it may sometimes be necessary to obtain a more accurate representation of the eye opening. As we have indicated before, such increased accuracy can be achieved simply by asking ABCD-L to use more bits to discretize the underlying circuit's signals. This is illustrated in Fig. 9, which depicts the eye diagram produced by 8-bit ABCD-L for the same RLGC chain. Indeed, as seen from the figure, the 8-bit ABCD-L model almost perfectly reproduces

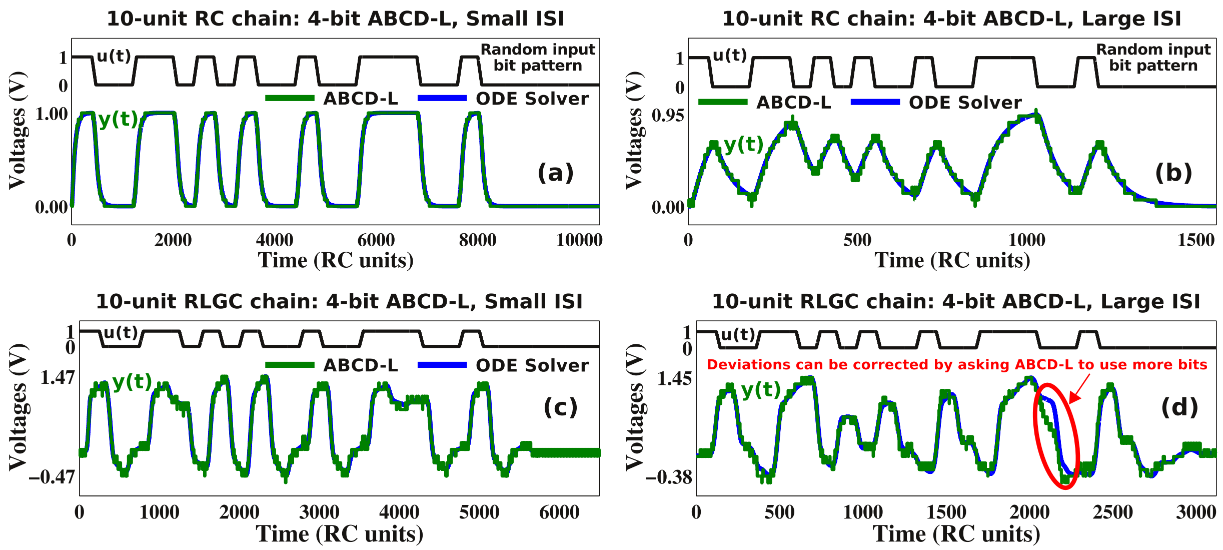


Fig. 6. Applying 4-bit and 8-bit ABCD-L to a 10-unit RC chain. The resulting Boolean models are simulated under conditions of both small ISI (parts (a), (c)) and large ISI (parts (b), (d)), and the Boolean models' predictions (the green waveforms) are compared against actual system responses (the blue waveforms), for a randomly generated input pattern (the black waveforms labelled $u(t)$).

the shape of the entire eye opening (as compared to the red dashed contour obtained by ODE simulation). This also confirms our assertion that ABCD-L can approximate LTI systems to any desired level of accuracy.

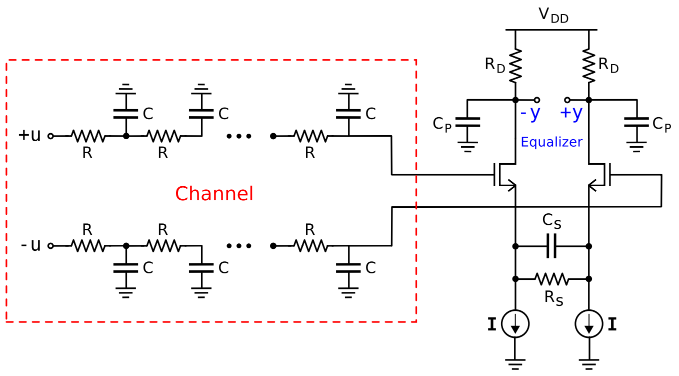


Fig. 10. LTI channel followed by a differential equalizer.

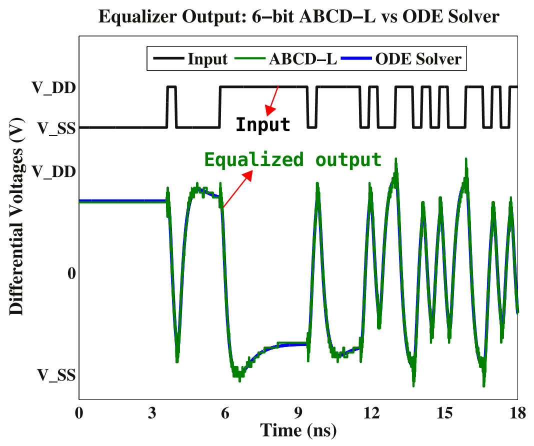


Fig. 11. ABCD-L accurately reproduces the time-domain continuous dynamics of the equalizer.

Having applied ABCD-L to RC/RLGC chains, we now consider a more complex example relevant to AMS-design: an LTI channel followed by an equalization circuit, as illustrated in Fig. 10. We note that, in this circuit, both the channel (modelled as an RC chain) and the equalizer use *differential signalling*, *i.e.*, the circuit's inputs and outputs are represented by the *difference* between two voltages (instead of a single voltage)[6]. The equalizer plays a critical role in this circuit: it *partially reverses* the distortion (ISI) produced by the channel, so that one can transmit bits across the channel at much higher speeds than would be possible otherwise. For example, if the channel's cut-off frequency is 1 GHz, then reliable transmission can happen only at bitrates at or below 1Gbps. However, if the combined "channel plus equalizer" system has an effective cut-off frequency at 3 GHz, then one can triple the throughput without suffering distortion.

We also note that the circuit in Fig. 10 is non-linear. Therefore, we apply ABCD-L not to the original circuit, but to a small-signal linearization of the original circuit around its quiescent operating point, using small-signal device models obtained from, *e.g.*, the book by Sedra and Smith [23].

Fig. 11 illustrates the application of 6-bit ABCD-L to the small-signal linearized "channel plus equalizer" circuit (henceforth simply referred to as "the system") above (with the channel being a 5-unit RC chain). The blue waveform of Fig. 11 was obtained by using an ODE solver to simulate the system, for a random bit pattern applied at the input (the black waveform). As before, we see from the figure that the Boolean model produced by ABCD-L is able to accurately duplicate the time-domain behaviour of the system.

For equalizers, an important AMS-relevant design metric is the eye diagram correction produced by the circuit. In typical AMS applications, the eye diagram at the input to the equalizer (*i.e.*, the channel output) has a very small or even non-existent eye opening (Fig. 12(a)). The equalizer offsets some of the ISI produced by the channel, which can considerably widen the eye opening; for example, Fig. 12(b) shows the eye diagram produced by a small-signal SPICE simulation of the above system, using SpiceOPUS [24]. Parts (c) and (d) of Fig. 12 depict the eye diagrams produced by applying 6-bit and 8-bit ABCD-L to the above "channel plus equalizer" system, overlaid on top of the (blue) SPICE eye diagram. As the figures show, the eye diagrams obtained from ABCD-L's Boolean models are able to accurately reproduce the eye diagram correction achieved by the equalizer. Thus, we have demonstrated that ABCD-L is a viable technique to Booleanize the continuous dynamics of AMS systems.

Having presented results pertaining to ABCD-L's accuracy, we now consider its computational efficiency. As outlined in §II, the Boolean models produced by ABCD-L lend themselves to efficient time-domain simulation carried out entirely in the discrete/logical domain, without having to solve differential equations. Even after taking into account the time taken to generate the ABCD-L models, ABCD-L can still be many times faster than conventional circuit simulation techniques like linear multi-step integration. This is illustrated in Fig. 13, which compares the total time taken by 4-bit, 5-bit, 6-bit, and 8-bit ABCD-L (total runtime includes pre-processing, model generation, simulation, and post-processing) against the time taken by Backward Euler integration, for simulating RC chains of various lengths on a long, randomly generated input bit pattern. As the figure shows, ABCD-L does offer a significant speedup advantage. Moreover, as the LTI system size increases, this advantage becomes even more pronounced[7]. In addition, as discussed in §II (and illustrated in the supplemental material), it is straightforward to integrate linear MOR techniques (*e.g.*, Arnoldi iteration [17], [18]) into

[6]Differential signalling has important advantages over single-ended signalling, including better noise resilience, improved resistance to external interference, *etc.*

[7]All the ABCD-L simulations of Fig. 13 have been carried out in C++, on a system equipped with a 6-core 3.2 GHz AMD® Phenom^{TM} II X6 1090T processor, and with a total of 16GB (shared) memory.

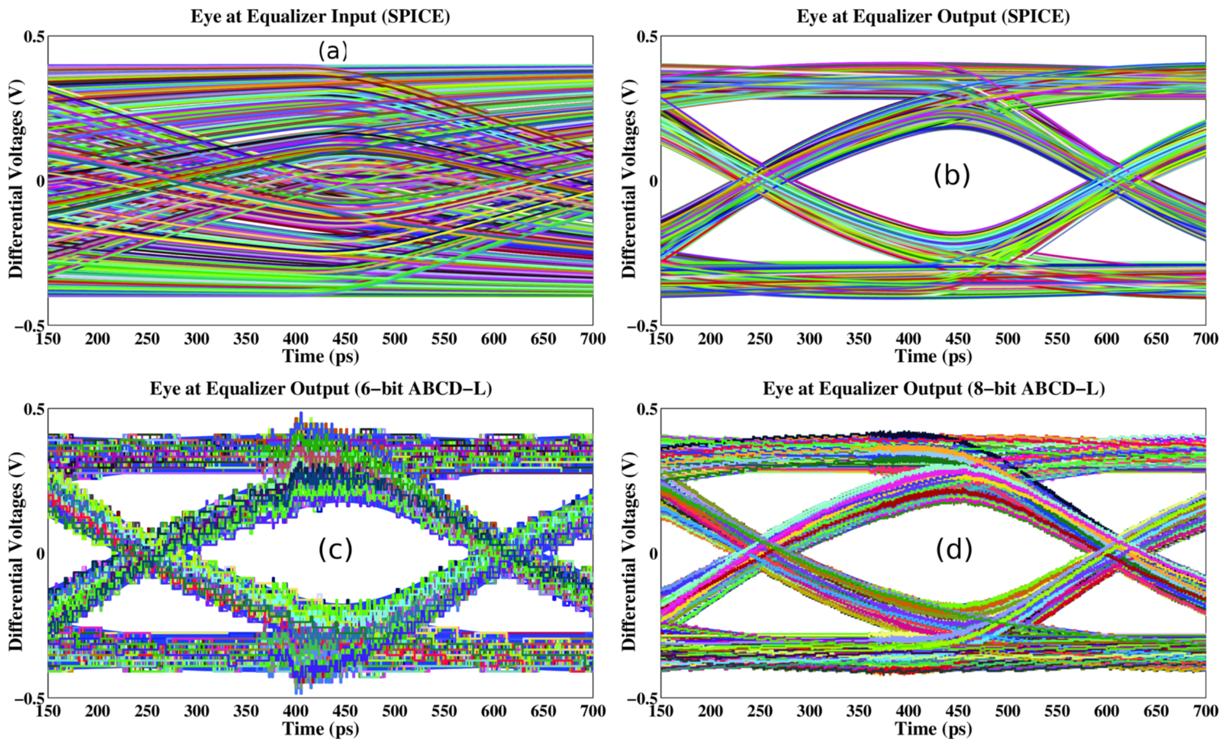


Fig. 12. ABCD-L accurately reproduces the entire shape of the eye diagram at the equalizer's output.

ABCD-L, which can further improve its runtime.

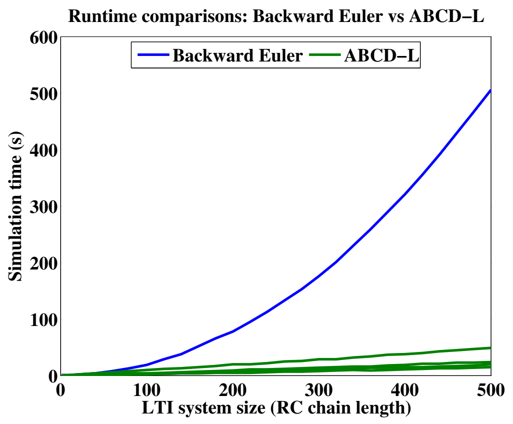


Fig. 13. ABCD-L can offer considerable simulation speed-up over traditional circuit simulation techniques like Backward Euler integration. The figure illustrates this for RC chains of varying length, and for 4-bit, 5-bit, 6-bit, and 8-bit ABCD-L.

IV. Summary, Conclusions, and Future Work

In this paper, we have developed and demonstrated ABCD-L, a technique that automatically produces Boolean approximations of continuous LTI systems, to any desired level of accuracy, in a completely scalable fashion. We have applied ABCD-L to representative LTI systems such as RC/RLGC chains, where it captures important analog effects like inter-symbol interference, ringing, *etc.* We have also demonstrated ABCD-L on a small-signal linearized "channel plus equalizer" circuit, where it is able to reproduce key design-relevant AMS metrics, including the eye diagram correction achieved by the circuit. Also, we have shown that ABCD-L generated models can offer significant simulation speed-up over conventional circuit simulation techniques like linear multi-step integration.

Through ABCD-L, we have demonstrated that systems specified in purely Boolean form have the ability to capture continuous-time LTI dynamics with excellent accuracy and scalability. However, it is important to develop this idea further, and bring ABCD-L to closure with formal verification methods, Boolean SAT solvers, *etc.* The first step in this regard is to develop techniques for efficiently synthesizing the combinational logic present in ABCD-L models. To this end, we are actively exploring new ways to exploit the structure of the underlying floating point computations, using specialized tools like ABC [4], to come up with compact, gate-level descriptions of ABCD-L models.

Our long-term future plans are shaped by our belief that ABCD-L possibly espouses a new route to the longstanding problem of SPICE-accurate AMS verification. By Booleanizing continuous-time LTI dynamics, ABCD-L capitalizes on the ability of existing Boolean/hybrid systems modelling/verification techniques to handle large numbers of Boolean/discrete variables. At the same time, ABCD-L attempts to steer clear of the main weakness of existing formal methods, namely, their scalability limitations while analyzing systems with more than a few continuous variables. Therefore, in future, we would like to work on (1) extending ABCD-L to Booleanize strongly non-linear systems as well (*e.g.*, mixers, oscillators, *etc.*), and (2) integrating ABCD-L with state-of-the-art techniques for Boolean and hybrid systems verification, to develop a new AMS verification methodology founded on Booleanizing continuous dynamics.

References

[1] K. V. Aadithya and J. Roychowdhury. DAE2FSM: Automatic generation of accurate discrete-time logical abstractions for continuous-time circuit dynamics. In *DAC '12: Proceedings of the* 49*th* *Design Automation Conference*, pages 311–316, 2012.
[2] G. Taylor. Future of analog design and upcoming challenges in nanometer CMOS. Keynote address at the 2010 International Conference on VLSI Design.
[3] R. Parker. Analog design challenges in the new era of process scaling. At the 2012 International Workshop on Design Automation for AMS Circuits (co-located with ICCAD).
[4] R. K. Brayton and A. Mishchenko. ABC: An academic industrial-strength verification tool. In *CAV '10: Proceedings of the 22nd International Conference on Computer Aided Verification*, pages 24–40, 2010.
[5] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. HyTech: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1):110–122, 1997.
[6] C. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91(7):986–1001, 2003.
[7] A. Chutinan and B. H. Krogh. Computational techniques for hybrid system verification. *IEEE Transactions on Automatic Control*, 48(1):64–75, 2003.
[8] G. Al-Sammane, M. H. Zaki, and S. Tahar. A symbolic methodology for the verification of AMS designs. In *DATE '07: Proceedings of the ACM Conference on Design, Automation and Test in Europe*, pages 249–254, 2007.
[9] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. *International Journal on Software Tools for Technology Transfer*, 10(3):263–279, 2008.
[10] S. Little. *Efficient Modeling and Verification of Analog/Mixed-Signal Circuits Using Labeled Hybrid Petri Nets*. PhD thesis, University of Utah, 2008.
[11] M. Althoff, A. Rajhans, B. H. Krogh, S. Yaldiz, X. Li, and L. Pileggi. Formal verification of Phase Locked Loops using reachability analysis and continuization. In *ICCAD '10: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 659–666, 2010.
[12] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, 2000.
[13] C. W. Gear. *Numerical initial value problem in ordinary differential equations*. Prentice Hall, Inc., 1971.
[14] B. Gustavsen and A. Semlyen. Rational approximation of frequency domain responses by vector fitting. *IEEE Transactions on Power Delivery*, 14(3):1052–1061, 1999.
[15] C. P. Coelho, J. Phillips, and L. M. Silveira. A convex programming approach for generating guaranteed passive approximations to tabulated frequency data. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(2):293–301, 2006.
[16] G. Strang. *Linear algebra and its applications*. Thomson, Brooks/Cole, 2006.
[17] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *The Quarterly of Applied Mathematics*, 9(1):17–29, 1951.
[18] W. H. A. Schilders, H. A. van der Vorst, and J. Rommes. *Model Order Reduction: Theory, research aspects and applications*, volume 13 of *Mathematics in Industry*. Springer Verlag, 2008.
[19] N. Brisebarre, F. De Dinechin, and J. M. Muller. Integer and floating-point constant multipliers for FPGAs. In *ASAP' 08: Proceedings of the* 19*th* *IEEE International Conference on Application Specific Systems, Architectures and Processors*, pages 239–244, 2008.
[20] P. K. Hanumolu, G. Y. Wei, and U. K. Moon. Equalizers for high-speed serial links. *International Journal of High Speed Electronics and Systems*, 15(2):429–458, 2005.
[21] J. A. Davis and J. D. Meindl. *Interconnect technology and design for gigascale integration*. Springer, Netherlands, 2003.
[22] G. Balamurugan, B. Casper, J. E. Jaussi, M. Mansuri, F. O'Mahony, and J. Kennedy. Modelling and analysis of high-speed I/O links. *IEEE Transactions on Advanced Packaging*, 32(2):237–247, 2009.
[23] A. S. Sedra and K. C. Smith. *Microelectronic circuits*. Oxford University Press, 2007.
[24] http://www.spiceopus.si/.

ABCD-L: Approximating Continuous Linear Systems Using Boolean Models (Supplement)

Abstract—**In this supplement, we provide additional context for ABCD-L and place our contributions in perspective, relative to the existing body of literature on topics like AMS modelling/verification, Boolean and hybrid systems frameworks, *etc.* Further, we demonstrate that ABCD-L can be applied in conjunction with Model Order Reduction (MOR) techniques, to Booleanize large LTI systems whose direct eigendecomposition may be computationally infeasible. For example, we combine ABCD-L with Arnoldi iteration based MOR to efficiently produce accurate Boolean models of a real-world power grid network (with 25849 nodes) obtained from a benchmark set made available by IBM. Due to space constraints, we were unable to include such material within our main manuscript.**

I. ABCD-L IN THE CONTEXT OF EXISTING FORMAL TECHNIQUES

Much of the existing body of work on the formal analysis and modelling of AMS systems has been carried out by the Boolean and hybrid systems verification communities. This literature is too vast to cover in full detail here; however, we will provide a brief overview highlighting the common features shared by existing approaches, and how ABCD-L complements them.

One trait that is shared by almost all existing formal verification systems is that they work with simplified behavioural models for AMS components – models that do not bear close resemblance to SPICE. Indeed, many general frameworks have been proposed for formal verification and reachability analysis of dynamical systems that involve both discrete and continuous variables; however, the verification involving continuous quantities often scales much more poorly than verification involving purely Boolean/discrete quantities. This limits the applicability of the proposed techniques/frameworks to behavioural models of AMS systems, rather than models that achieve SPICE-level accuracy.

For example, consider the work by Ghosh and Vemuri [1], who applied the PVS proof checking tool to simplified analog circuit models (*e.g.*, using idealized OpAmps, transistors with constant transconductance, *etc.*). While this was an important step towards AMS verification, its range of applications was limited by computational challenges arising from the need to formally verify arithmetic over real numbers.

Due to the inherent scalability limitations of verifying continuous systems, many other techniques were developed to abstract analog dynamics using highly simplified behavioural models, carefully tailored to specific application domains/circuit classes. For example, Hanna and others [2], [3] developed new approaches to formally verify digital circuits suffering from analog non-idealities.

There is also another class of AMS verification approaches; these methods try to partition the continuous analog state space of voltages and currents into discrete domains, and the idea is to encode transitions between these domains using Boolean data structures like Finite State Machines, Binary Decision Diagrams, *etc.* For example, the work by Kurshan and Macmillan [4], Hedrich et. al. [5], [6], *etc.* fall into this category. The formal verification and model checking of such abstractions can be carried out using either off-the-shelf techniques with only small modifications (*e.g.*, as done by Kurshan), or by specially augmenting existing CTL model-checking tools with extra AMS-relevant features (as espoused by Hedrich and others). However, in spite of researchers' best efforts along these lines, their techniques were scalable only to small designs (*e.g.*, a single gate [4], or a small tunnel diode [5]), and were also limited in their power to model real-world analog phenomena that were of interest to AMS designers.

With a view to scaling up verification techniques to real AMS designs, several new modelling frameworks, with associated verification methodologies, were introduced. Together, these fall under the umbrella of "hybrid systems verification", a topic that has been extensively studied in the literature, and mathematically formalised by researchers like Alur, Nerode, and Henzinger [7]–[10].

The above modelling/formalisation efforts were critical because many reachability questions on general hybrid systems are, in fact, undecidable. Therefore, for AMS verification, it is necessary to restrict one's domain to classes of hybrid systems that are known to be decidable; the above formalisation efforts helped develop a strong *theory of hybrid systems*, that enabled researchers to ask more meaningful questions, which in turn enabled new advances in reachability analysis/model checking of AMS systems.

For instance, Al-Sammane et. al. used recurrence relations and difference equations to simplify analog blocks [11], which were later verified using interval arithmetic and Taylor approximations [12]. This methodology was successfully used to check the stability of a third order $\Delta\Sigma$ modulator (modelled behaviourally). Also of significance is the d/dt tool, developed exclusively for model checking continuous linear systems [13], although limited to rather small system sizes.

Other novel hybrid systems modelling frameworks (that restrict themselves to decidable hybrid systems classes) for AMS include: guarded state machines as applied to flash memories, verified using the ACL2 theorem prover [14], linear hybrid automata verified using tools like HyTech [15] and Phaver [16], Polyhedral invariant hybrid automata, verified using flowpipes and the theory of quotient transition systems, by tools like Checkmate [17], labelled hybrid petri-nets, verified using difference bound matrices as part of the LEMA toolkit developed by Myers et. al. [18]–[21], and many more that we do not mention here due to space constraints. In addition, several new algorithmic refinements have improved the accuracy and efficiency of hybrid systems' reachability analysis. These include zonotopes [22], hybrid restriction diagrams [23], and other over-approximation techniques (*e.g.*, using support functions [24], continuization methods [25], *etc.*).

Thus, much effort has been devoted to developing and fine-tuning Boolean and hybrid systems modelling frameworks and verification engines. However, it is our belief that the question of *SPICE-accurate modelling*, which ensures that the circuit models used by verification engines actually reflect underlying analog reality, has not received adequate scrutiny. Without SPICE-accurate modelling, the predictions made by verification engines are questionable and dangerous for designers to rely on. Indeed, this is an important reason why the prevailing practice amongst AMS designers today is to carry out time-consuming SPICE simulations rather than place their trust in AMS verification tools.

To overcome such "designer skepticism", we believe that it is necessary to significantly scale up existing hybrid systems techniques, so that they embrace SPICE-accurate models even for large AMS designs. However, this creates scalability problems. To circumvent such scalability issues, we suggest that continuous variables (which introduce major computational challenges that are orders of magnitude more severe than purely discrete/Boolean variables) be used as sparingly as possible in the modelling of AMS components.

Thus, in our view, there is a need to develop new techniques that accurately capture the behaviour of continuous systems using purely discrete/Boolean variables, even though, in theory, existing hybrid systems approaches can represent and reason about continuous quantities. Armed with such techniques, we believe that much of an AMS system's behaviour will be representable using "cheap" purely discrete/Boolean variables, which frees up the "precious" continuous variables for use only when absolutely necessary. This may help arrest the scalability issues inherent to existing hybrid systems approaches, and may one day enable the verification of large AMS systems at or near SPICE-level accuracy.

We view ABCD-L as a step in the above direction, which addresses the problem of Booleanizing analog dynamics, for LTI systems in particular. In future, we would like to extend ABCD-L to non-linear systems as well, and to integrate ABCD-L with existing Boolean and hybrid systems frameworks for AMS verification, to expand the scope of existing techniques to handle much larger systems than they can do so at present. This is our hope for the future of AMS verification, and is also the larger context behind ABCD-L.

II. ABCD-L COMBINED WITH MODEL ORDER REDUCTION

As described in §II of the main manuscript, ABCD-L requires eigendecomposition of LTI systems as part of the Booleanization process. However, as LTI system size increases, eigendecomposition can quickly become computationally infeasible. To address this problem, we suggest an approach involving Model Order Reduction (MOR). The idea is to first obtain a Reduced Order Model (ROM) of the original LTI system; many well-established techniques are available for this, including explicit moment matching methods such as Asymptotic Waveform Evaluation (AWE) [26], implicit Krylov subspace methods such as congruent transformation [27], Padé approximation via the Lanczos method [28], guaranteed-stable methods based on Arnoldi iteration [29], [30], *etc.* The next step is to apply ABCD-L to the ROM, which is typically much smaller than the original LTI system, and therefore not a barrier for eigendecomposition.

We now apply the above approach to a real-world power grid network, obtained from a benchmark set made available by IBM [31], [32]. This LTI network has 25849 nodes, making eigenanalysis slow and impractical. Therefore, we first carry out Arnoldi iteration [33] based MOR to reduce this system to a more manageable size. Indeed, as we show below, a ROM of size $\sim$20 suffices to capture the dynamics of this system for most frequencies of interest. We then Booleanize the ROM using ABCD-L, and show that the resulting Boolean model is able to accurately reproduce the behaviour of the original system, including such attributes as the power grid's voltage swings in the ground plane and in the V_{DD} plane.

A. Arnoldi ROMs for the power grid

Given an LTI system L_{orig} of size n, and a desired ROM size p (where $p \ll n$), the method of Arnoldi iteration produces an LTI system L_{ROM} of size p, that approximates the behaviour of the original system L_{orig}. The key idea behind Arnoldi MOR is to *match moments*, *i.e.*, the reduced order model L_{ROM} is constructed in such a way that the first p moments of its transfer function are identical to those of the original system L_{orig}. In addition to being fast, Arnoldi iteration has favorable numerical properties (in the context of fixed-precision computation), as opposed to other techniques like Padé approximation [33], [34].

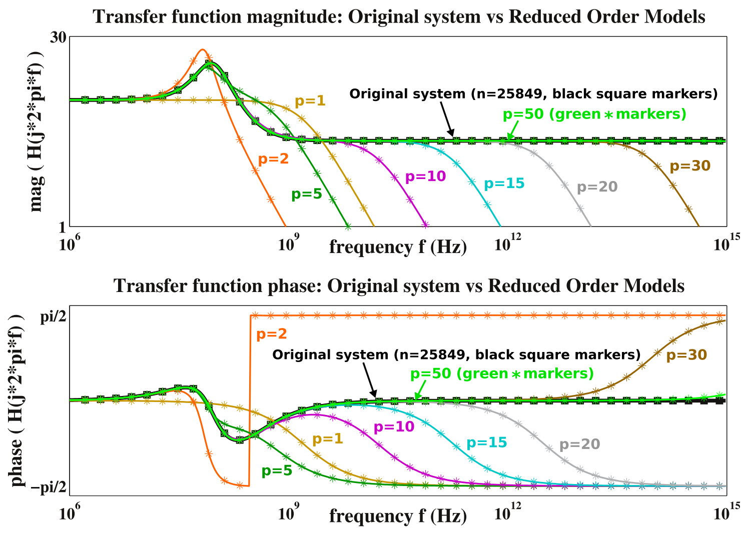


Fig. 1. Arnoldi ROM applied to the IBM power grid network. As ROM size p increases, the reduced order models produced by Arnoldi iteration become better and better approximations of the original system.

To determine a suitable ROM size p for the IBM power grid, Fig. 1 compares the LTI frequency-domain transfer function (both magnitude and phase) of the original power grid against those of several different Arnoldi ROMs, corresponding to different sizes p, over a wide frequency range (1 Megahertz to 1000 Terahertz). The black waveform (with black square markers) represents the original system's transfer function, while the '*' marked color waveforms correspond to the Arnoldi ROMs (with sizes as labelled in the figure). The figure clearly brings out the effectiveness of Arnoldi ROM for this network; for example, even though the original system is of size 25849, a ROM of size $\sim$20 suffices to accurately capture the system's behaviour for input excitations upto 100GHz. Therefore, an Arnoldi ROM of size $\sim$20 is more than adequate for most typical power grid applications[1].

[1]We note that Fig. 1 depicts the transfer function for only one output node of the power grid. In reality, the transfer functions corresponding to all output nodes (both in the power plane and in the ground plane) must be examined before concluding that ROM size $p = 20$ is adequate. For the given network, we have confirmed that this is indeed the case; however, due to space constraints, we do not include all the transfer function plots here.

B. ABCD-L + Arnoldi MOR applied to the power grid

We now apply ABCD-L to produce purely Boolean approximations of the ROMs generated above. As noted earlier, these ROMs are much smaller systems compared to the original power grid, and hence pose no difficulty for eigenanalysis.

Fig. 2 shows that the Boolean models produced by ABCD-L are able to accurately reproduce the transient dynamics of the Arnoldi ROMs generated for the IBM power grid. Parts (a), (b), (c), and (d) of the figure correspond to ROM sizes 5, 8, 10, and 20 respectively. In each case, the power grid was excited by the same input waveform $u(t)$ – a superposition of two damped sinusoidal excitations at 10GHz, one positive and the other negative (see Fig. 3).

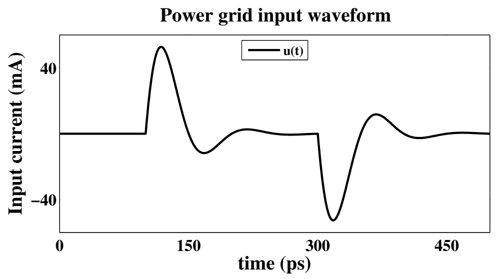


Fig. 3. Input current waveform applied to the IBM power grid: a superposition of two damped sinusoidal excitations at 10GHz, one positive and the other negative.

Each part of Fig. 2 depicts the following: (1) two output waveforms (one output from the power plane and one from the ground plane) produced by the original power grid (in dark gray), (2) the same outputs, as predicted by the respective reduced order model (in blue), and (3) the outputs as predicted by 5-bit ABCD-L applied to the corresponding ROM (in green).

From the figure, it is clear that the Boolean models generated by ABCD-L always closely reproduce the corresponding ROM's response. Moreover, as the ROM size increases, this response in turn becomes a very good approximation to the response of the original power grid system, both in the power plane and in the ground plane.

Furthermore, as expected, the combination of ABCD-L with Arnoldi MOR resulted in significant computational savings over direct eigendecomposition of the original system. For instance, even with $p = 100$, the Arnoldi step took only about 10 minutes, on a 64-bit Linux machine equipped with a 3.2GHz AMD® Phenom$^{\text{TM}}$ II X6 1090T processor and 16GB RAM. Moreover, once the Arnoldi iteration was completed, the time required for ABCD-L (including pre-processing, model generation, transient simulation, and post-processing) was well under a minute. This shows that ABCD-L, in conjunction with MOR, is indeed a viable technique that can be applied to accurately Booleanize even large LTI systems.

References

[1] A. Ghosh and R. Vemuri. Formal verification of synthesized analog designs. In *ICCD '99: Proceedings of the IEEE International Conference on Computer Design*, pages 40–45, 1999.

[2] K. Hanna. Reasoning about real circuits. In *Proceedings of the 7th International Workshop on Higher Order Logic Theorem Proving and Its Applications*, pages 235–253, 1994.

[3] K. Hanna. Reasoning about analog-level implementations of digital systems. *Formal Methods in System Design*, 16(2):127–158, 2000.

[4] R. P. Kurshan and K. L. McMillan. Analysis of digital circuits through symbolic reduction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(11):1356–1371, 1991.

[5] W. Hartong, L. Hedrich, and E. Barke. Model checking algorithms for analog verification. In *DAC '02: Proceedings of the 39th annual ACM Design Automation Conference*, pages 542–547, 2002.

[6] S. Steinhorst and L. Hedrich. Model checking of analog systems using an analog specification language. In *DATE '08: Proceedings of the ACM Conference on Design, Automation and Test in Europe*, pages 324–329, 2008.

[7] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. *Hybrid Systems*, pages 209–229, 1993.

[8] A. Nerode and W. Kohn. Models for hybrid systems: Automata, topologies, controllability, observability. *Hybrid Systems*, pages 317–356, 1993.

[9] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[10] T. A. Henzinger. The theory of hybrid automata. In *LICS '96: Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science*, pages 278–292, 1996.

[11] G. Al-Sammane, M. H. Zaki, and S. Tahar. A symbolic methodology for the verification of AMS designs. In *DATE '07: Proceedings of the ACM Conference on Design, Automation and Test in Europe*, pages 249–254, 2007.

[12] M. H. Zaki, G. Al-Sammane, S. Tahar, and G. Bois. Combining symbolic simulation and interval arithmetic for the verification of AMS designs. In *FMCAD '07: Formal Methods in Computer Aided Design*, pages 207–215, 2007.

[13] E. Asarin, T. Dang, and O. Maler. d/dt: A verification tool for hybrid systems. In *CDC '01: Proceedings of the 40th IEEE Conference on Decision and Control*, pages 2893–2898, 2001.

[14] S. Ray, J. Bhadra, T. Portlock, and R. Syzdek. Modeling and verification of industrial flash memories. In *ISQED '10: Proceedings of the 11th International Symposium on Quality Electronic Design*, pages 705–712, 2010.

[15] T. A. Henzinger, P. H. Ho, and H. Wong-Toi. HyTech: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1):110–122, 1997.

[16] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. *International Journal on Software Tools for Technology Transfer*, 10(3):263–279, 2008.

[17] A. Chutinan and B. H. Krogh. Computational techniques for hybrid system verification. *IEEE Transactions on Automatic Control*, 48(1):64–75, 2003.

[18] S. Little, N. Seegmiller, D. Walter, C. Myers, and T. Yoneda. Verification of ams circuits using labeled hybrid petri nets. In *ICCAD '06: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 275–282, 2006.

[19] S. Little, D. Walter, K. Jones, and C. Myers. AMS circuit verification using models generated from simulation traces. *Automated Technology for Verification and Analysis*, pages 114–128, 2007.

[20] S. Little. *Efficient Modeling and Verification of Analog/Mixed-Signal Circuits Using Labeled Hybrid Petri Nets*. PhD thesis, University of Utah, 2008.

[21] S. Batchu. Automatic Extraction of Behavioral Models from Simulations of Analog/Mixed-Signal (AMS) Circuits. Master's thesis, University of Utah, 2010.

[22] M. Althoff, O. Stursberg, and M. Buss. Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes. *Nonlinear Analysis: Hybrid Systems*, 4(2):233–249, 2010.

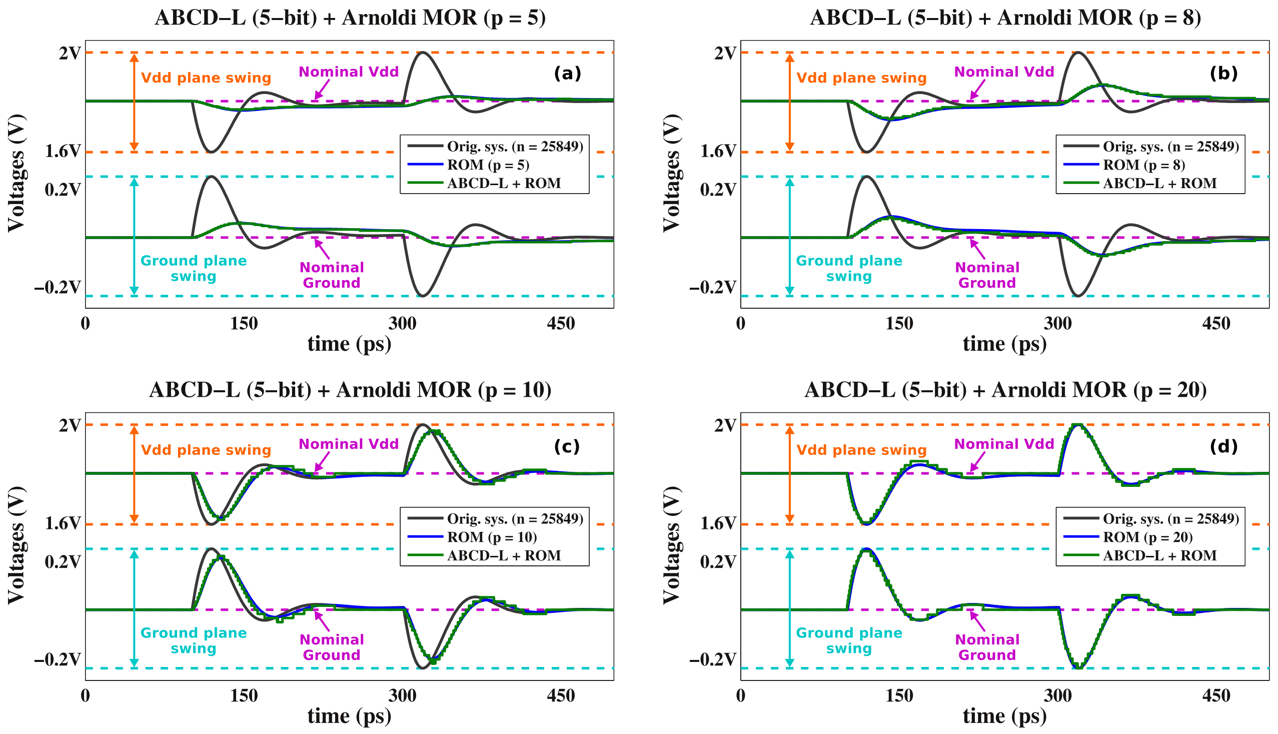


Fig. 2. ABCD-L plus Arnoldi MOR applied to the IBM power grid, for different ROM sizes p. The plots depict the network's response to the input $u(t)$ of Fig. 3. In each case, it is seen that the Boolean model generated by ABCD-L closely approximates the ROM's response. And as ROM size increases, this response becomes a very good approximation to the response of the original power grid system.

[23] F. Wang. Symbolic parametric safety analysis of linear hybrid systems with BDD-like data-structures. *IEEE Transactions on Software Engineering*, 31(1):38–51, 2005.

[24] C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *CAV '09: Proceedings of the International Conference on Computer Aided Verification*, pages 540–554, 2009.

[25] M. Althoff, A. Rajhans, B. H. Krogh, S. Yaldiz, X. Li, and L. Pileggi. Formal verification of Phase Locked Loops using reachability analysis and continuization. In *ICCAD '10: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 659–666, 2010.

[26] L. T. Pillage and R. A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):352–366, 1990.

[27] K. J. Kerns, I. L. Wemple, and A. T. Yang. Stable and efficient reduction of substrate model networks using congruence transforms. In *ICCAD '95: Proceedings of the IEEE/ACM International Conference on Computer-Aided design*, pages 207–214, 1995.

[28] P. Feldmann and R. W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(5):639–649, 1995.

[29] L. Silveira, M. Kamon, I. Elfadel, and J. White. A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits. *Computer Methods in Applied Mechanics and Engineering*, 169(3):377–389, 1999.

[30] A. Odabasioglu, M. Celik, and L. T. Pileggi. Prima: Passive reduced-order interconnect macromodeling algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(8):645–654, 1998.

[31] S. R. Nassif. Power grid analysis benchmarks. In *ASPDAC '08: Proceedings of the* 13th *IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 376–381, 2008.

[32] Download link for the IBM power grid benchmark set: http://dropzone.tamu.edu/~pli/PGBench/.

[33] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *The Quarterly of Applied Mathematics*, 9(1):17–29, 1951.

[34] W. H. A. Schilders, H. A. van der Vorst, and J. Rommes. *Model Order Reduction: Theory, research aspects and applications*, volume 13 of *Mathematics in Industry*. Springer Verlag, 2008.