# Transient Adjoint DAE Sensitivities: a Complete, Rigorous, and Numerically Accurate Formulation

Naomi Sagan and Jaijeet Roychowdhury
Department of Electrical Engineering and Computer Sciences, University of California, Berkeley
{naomi.sagan,jr}@berkeley.edu

***Abstract*—Almost all practical systems rely heavily on physical parameters. As a result, parameter sensitivity, or the extent to which perturbations in parameter values affect the state of a system, is intrinsically connected to system design and optimization. We present TADsens, a method for computing the parameter sensitivities of an output of a differential algebraic equation (DAE) system. Specifically, we provide rigorous, insightful theory for adjoint sensitivity computation of DAEs, along with an efficient and numerically well-posed algorithm implemented in Berkeley MAPP. Our theory and implementation advances resolve long-standing issues that have impeded adoption of adjoint transient sensitivities in circuit simulators for over 5 decades. We present results and comparisons on two nonlinear analog circuits. TADsens is numerically well posed and accurate, and faster by a factor of 300 over direct sensitivity computation on a circuit with over 150 unknowns and 600 parameters.**

## I. Introduction

The operation of every circuit depends strongly on parameters such as resistance and capacitance values, transistor dimensions, MOSFET threshold voltages, doping concentrations, *etc.* In modern ICs, where transistor dimensions are on the order of atoms, parameter variations and their effects on performance are of serious concern during design and fabrication. Transient parameter sensitivities, *i.e.*, the effect of parameter fluctuations on a system's waveforms, identify which parameters have the greatest impact on system outputs. This information is essential for designing systems that are robust to parameter variations.

Implicitly or explicitly, simulators pose circuits mathematically as Differential Algebraic Equations (DAEs) [1, 2]. In contrast to Ordinary Differential Equations (ODEs), which involve differential components in each equation, DAEs include purely algebraic equations, *i.e.*, equations lacking differential terms. Indeed, equations that appear superficially to be ODEs may really be DAEs; in most circuit applications, it is difficult if not impossible to separate algebraic components from "genuinely differential" equations. Transient sensitivity techniques for ODEs, which are simpler than those for DAEs in both concept and implementation, do not work correctly for DAEs.

A straightforward way to compute transient sensitivities is to derive a linear time-varying DAE directly in sensitivity unknowns and to solve it via transient analysis. This method, known as "direct sensitivity computation", becomes highly inefficient for systems with many parameters.[1] It involves performing transient analysis on a DAE in a large, dense, *matrix* unknown.[2] Since real-life circuits typically have very many parameters, direct transient sensitivity calculation is often too memory- and computation-intensive to be practical.

It is possible, however, to circumvent this expensive computation if the sensitivities of only a *few system outputs*, instead of those of the entire state vector of the DAE, are needed. In almost all practical applications, this is sufficient—only a few outputs are of interest for sensitivities. Parameter sensitivities of a single scalar output can be efficiently computed using a different approach, called "adjoint sensitivity computation". The essential idea is to use adjoints of linear operators [3], which fundamentally conserve inner products, to isolate the parametric sensitivities of a single output at any time of interest. The key to the efficiency of adjoint methods is that a quantity necessary for this inner product, the *adjoint sensitivity function* (ASF), can be found by solving a time-varying differential equation (the adjoint DAE) in a *vector* unknown, avoiding the large, dense matrix unknown of the direct method.

Work on adjoint sensitivity computation has spanned 5 decades. [4] provided "adjoint networks" for a few circuit element types, but applying this concept in circuit simulators soon revealed tricky theoretical and numerical issues; as a result, transient adjoint sensitivities were avoided in most simulators. For example, adjoint computations require impulsive ($\delta$-function) inputs, which being infinite-valued are not well suited for numerical computation. This was one of the reasons why [5] chose to implement direct sensitivities in **TI**SPICE, noting it to be "much more straightforward" than adjoint. [6] implemented adjoint sensitivities in SPECS, a specialized IBM simulator using piecewise-constant circuit elements and event-driven time stepping; but the adjoint techniques used were similarly specialized and do not port to standard circuit simulators. [7], noting that "explicitly constructing" the adjoint DAE system is "quite nontrivial", transformed the direct method into a large chained matrix product. This was achieved by discretizing the direct DAE in time and unrolling transient time stepping; matrix adjoints (*i.e.*, transposes) were then applied for computational efficiency. More recently, [8] followed a similar approach, with the unrolled matrix products represented as a directed acyclic graph (DAG) that is traversed forward and backward for direct and unrolled-matrix-adjoint sensitivities, respectively. Neither method has addressed unresolved theoretical and numerical questions in transient adjoint sensitivities. [9, 10], which employed Lagrange multipliers to formulate adjoints, considered special DAE formulations that are different from those commonly used in circuit simulators. In [9, 10], treatment of DAE forms suited for circuits is limited to outputs that are integrals of circuit waveforms with finite kernels (single-timepoint outputs cannot be captured in this form). Moreover, [9, 10] do not identify or address the issue of impulsive components in the ASF—which, as we show here, can be present, causing numerical problems unless treated specially. Note that for ODEs, [11] showed that impulsive inputs to the adjoint equation can be handled by (analytical) integration over an infinitesimal interval, reducing them to finite-valued initial conditions that are perfectly amenable numerically; however, [11] does not consider DAEs. In short, key theory and implementation questions for transient adjoint sensitivities have remained unanswered.

In this paper, we provide a comprehensive exposition of the core questions in transient adjoint sensitivity computation, along with answers that are not only theoretically rigorous but also well suited for numerical implementation in circuit simulators. We make the following contributions:

1) We provide a clear, from-first-principles derivation[3] of transient adjoint DAE sensitivities that reveals under what conditions the adjoint DAE is useful, why it must be integrated backwards, *etc.* In particular, we show why impulsive inputs are needed for the adjoint DAE, and bring out that the ASF (solution of the adjoint

[1] This is illustrated in Sec. III for a 51-stage ring oscillator circuit.

[2] The unknown is the sensitivity matrix, with rows corresponding to DAE unknowns and columns to parameters.

[3] without using Lagrange multipliers [10] or Tellegen's theorem [4].

DAE) can have an impulsive component. Attempting to compute the ASF by applying numerical methods blindly will, therefore, result in severe errors. We show, however, that the ASF can be separated analytically into impulsive and finite components, and both components can be computed independently using well-posed numerics. The presence of the impulsive component in the ASF makes intuitive sense for DAEs, which contain algebraic equations; *e.g.*, a linear algebraic equation with an impulsive input will have a impulsive solution. Separating out and solving for the impulsive component involves finding the null space of a rank-deficient[4] DAE Jacobian matrix.

2) Further, we show that even after the impulsive component of the ASF has been separated out, the remaining finite-valued ASF component can be discontinuous at the initial timepoint due to the presence of algebraic equations in DAEs. The discontinuity leads to numerical inaccuracies in a post-processing step, needed for computing output sensitivities, that involves integrating the ASF. We provide simple means to circumvent this discontinuity and resultant inaccuracy.
3) We demonstrate our method, dubbed **TADsens** (Transient Adjoint DAE Sensitivities), on two circuits whose operation is fundamentally nonlinear (hence more challenging for simulators): a BJT Schmitt trigger circuit featuring hysteresis, and a 51-stage MOS ring oscillator circuit generating self-sustaining oscillations. On the latter, we use **TADsens** to find parametric sensitivities of the oscillation period. We provide comparisons against direct sensitivities, obtaining speedups of more than 2 orders of magnitude for the larger circuit. We also compare **TADsens** against analytically-derived sensitivities for a small, illustrative example.
4) We have implemented **TADsens** in Berkeley MAPP [12, 13], an open-source prototyping simulator, primarily because of its modular algorithm structuring and the powerful mathematical primitives available in MATLAB®. These features facilitated a compact, readable implementation that explicitly follows the theory and equations in this paper. We plan to release our implementation as open source, enabling our method to be easily ported to other simulators, including commercial ones.

In Section II-A, we state in concrete terms the problem setup and desired outputs of sensitivity computations, and examine the direct method. In Sections II-B and II-C, we apply adjoint operators to sensitivity computation, explaining the necessity of a $\delta$-function input into the adjoint DAE. We then derive the impulsive and finite-valued components of the ASF in Section II-D, via an infinitesimal integration. In Section II-E, we address the possible discontinuity of the ASF solution at the first timestep, as well as a method to mitigate the resulting numerical inaccuracies. We summarize a step-by-step procedure for computing sensitivities via **TADsens**, and compare the runtime of the direct and adjoint methods in Section II-F. In Section II-G, we examine adjoint DAEs for ODE and purely Algebraic Equation systems, revealing useful intuition about the two components of the ASF. In Section III, we detail the results of applying **TADsens** to three circuits, demonstrating that it is accurate and more efficient than direct computation.

# II. Theory

## A. Problem Definition and Direct Method

Consider a DAE in standard form,

$$\frac{d}{dt}\vec{q}(\vec{x}(t),\vec{p})+\vec{f}(\vec{x}(t),\vec{p})+\vec{b}(t)=\vec{0}, \tag{1}$$

with initial condition $\vec{x}(0)=\vec{x}_0$.[5] The functions $\vec{q}(\vec{x}(t),\vec{p})\in\mathbb{R}^n$ and $\vec{f}(\vec{x}(t),\vec{p})\in\mathbb{R}^n$, and therefore the solution $\vec{x}(t)\in\mathbb{R}^n$, depend on a set of parameters $\vec{p}\in\mathbb{R}^{n_p}$.

Given a nominal set of parameters, $\vec{p}_{\text{nom}}$ and input $\vec{b}_{\text{nom}}(t)$, we can solve the DAE using Linear Multi-Step (or LMS) methods [14] for nominal solution $\vec{x}_{\text{nom}}(t)$. From there, our goal is to find the parameter sensitivity matrix of the DAE solution,

$$\mathbf{M}(t)=\left.\frac{d\vec{x}}{d\vec{p}}\right|_{\vec{x}_{\text{nom}}(t),\vec{p}_{\text{nom}}}\in\mathbb{R}^{n\times n_p}. \tag{2}$$

The direct sensitivity computation for $\mathbf{M}(t)$ is derived below.

1) Solve for $\vec{x}_{\text{nom}}(t)$ over $t\in[0,T]$ for some end time $T$, nominal parameters $\vec{p}_{\text{nom}}$, and input $\vec{b}_{\text{nom}}(t)$. For each timestep of the simulation, compute Jacobian matrices
$$\mathbf{C}(t)=\frac{\partial\vec{q}}{\partial\vec{x}},\mathbf{G}(t)=\frac{\partial\vec{f}}{\partial\vec{x}},\mathbf{S_q}(t)=\frac{\partial\vec{q}}{\partial\vec{p}},\mathbf{S_f}(t)=\frac{\partial\vec{f}}{\partial\vec{p}}, \tag{3}$$
evaluated at $\vec{x}_{\text{nom}}(t)$ and $\vec{p}_{\text{nom}}$.
$\mathbf{C}(t)$ and $\mathbf{G}(t)$ are $\in\mathbb{R}^{n\times n}$ and $\mathbf{S_q}(t)$ and $\mathbf{S_f}(t)$ are $\in\mathbb{R}^{n\times n_p}$.
2) Rewrite (1), defining $\vec{p}=\vec{p}_{\text{nom}}+\Delta\vec{p}$ and $\vec{x}(t)=\vec{x}_{\text{nom}}(t)+\Delta\vec{x}(t)$, where $\Delta\vec{p}$ is a perturbation of the parameters from the nominal values, and $\Delta\vec{x}(t)$ is the corresponding deviation of $\vec{x}(t)$ from the nominal solution.
$$\frac{d}{dt}\vec{q}(\vec{x}_{\text{nom}}(t)+\Delta\vec{x}(t),\vec{p}_{\text{nom}}+\Delta\vec{p}) + \vec{f}(\vec{x}_{\text{nom}}(t)+\Delta\vec{x}(t),\vec{p}_{\text{nom}}+\Delta\vec{p})+\vec{b}_{\text{nom}}(t)=\vec{0}. \tag{4}$$
We can then linearize the DAE around $\vec{x}_{\text{nom}}$, $\vec{p}_{\text{nom}}$:
$$\frac{d}{dt}\left(\mathbf{C}(t)\Delta\vec{x}(t)+\mathbf{S_q}(t)\Delta\vec{p}\right)+\mathbf{G}(t)\Delta\vec{x}(t)+\mathbf{S_f}(t)\Delta\vec{p}\approx\mathbf{0}. \tag{5}$$
As $\Delta\vec{p}$ approaches $\vec{0}$, (5) approaches a strict equality and $\Delta\vec{x}(t)$ approaches $\mathbf{M}(t)\Delta\vec{p}$. So, taking the limit as $\Delta\vec{p}\to\vec{0}$, we get the following linear time-varying DAE:
$$\frac{d}{dt}\left(\mathbf{C}(t)\mathbf{M}(t)+\mathbf{S_q}(t)\right)+\mathbf{G}(t)\mathbf{M}(t)+\mathbf{S_f}(t)=\mathbf{0}, \tag{6}$$
with initial condition $\mathbf{M}(0)=\mathbf{0}$.
3) Solve (6) over $t\in[0,T]$ running a transient simulation for each column of $\mathbf{M}(t)$ separately.

This method, though straightforward, is computationally inefficient. We must solve $n_p$ separate DAEs of size $n$, which results in a runtime complexity of $O(nn_pT)$. The memory complexity is also $O(nn_pT)$, as the solution to (6) is commonly dense. For most physical systems, $n_p\gg n$, making a complexity of $O(nn_pT)$ nonviable in practice.

## B. Applying Adjoint Operators to Parameter Sensitivities

In almost all circuit applications, we do not use all entries of the sensitivity matrix $\mathbf{M}(t)$ over all $t\in[0,T]$. We instead focus on a single scalar output, $o(t)=\vec{c}^*\vec{x}(t)$, where $\vec{c}^*$ represents the conjugate transpose of output selection vector $\vec{c}\in\mathbb{R}^n$. Using adjoint operators, we can efficiently compute

$$\vec{m}^*(T)=\vec{c}^*\mathbf{M}(T), \tag{7}$$

the sensitivities of output $o(t)$ at time $t=T$.

The adjoint of linear operator $\mathscr{L}\{\vec{u}(t)\}=\vec{x}(t)$, for input $\vec{u}(t)$ and output $\vec{x}(t)$, is defined as $\mathscr{L}^\dagger\{\vec{y}(t)\}=\vec{z}(t)$, such that

$$\langle\vec{x}(t),\vec{y}(t)\rangle=\langle\vec{u}(t),\vec{z}(t)\rangle. \tag{8}$$

To apply adjoints to sensitivity computation, let us define our linear operator of interest to be $\mathscr{L}\{\vec{u}(t)\}=\Delta\vec{x}(t)$, where

$$u(t)=-\mathbf{S}(t)\Delta\vec{p}=-\left(\frac{d}{dt}\mathbf{S_q}(t)+\mathbf{S_f}(t)\right)\Delta\vec{p}. \tag{9}$$

This operator can be derived by rearranging terms of (5) to get

$$\frac{d}{dt}\mathbf{C}(t)\Delta\vec{x}(t)+\mathbf{G}(t)\Delta\vec{x}(t)=-\mathbf{S}(t)\Delta\vec{p}=\vec{u}(t). \tag{10}$$

Choose $\vec{y}(t)$, the input to the adjoint operator $\mathscr{L}^\dagger\{\vec{y}(t)\}=\vec{z}(t)$, to be $\vec{c}\delta(t-T)$.[6] The output $\vec{z}(t)$ of the adjoint is known as the *adjoint sensitivity function* (ASF).

Applying the sifting property of the Dirac $\delta$ [15] to (8), we select the sensitivities of $o(t)$ at a specific time $T$:

$$\begin{aligned}\int_A^B\vec{c}^*\Delta\vec{x}(t)\delta(t-T)dt&=-\int_A^B\vec{z}^*(t)\mathbf{S}(t)\Delta\vec{p}dt\\ \vec{c}^*\Delta\vec{x}(T)&=-\Delta\vec{p}\int_A^B\vec{z}^*(t)\mathbf{S}(t)dt.\end{aligned} \tag{11}$$

[4] Rank deficiency stems from the presence of algebraic equations.

[5] For this paper, we assume that neither $\vec{x}_0$ nor $\vec{b}(t)$ depend on $\vec{p}$.

[6] $\delta(t)$ is the Dirac $\delta$ function.

The upper bound of the integrals must satisfy $B > T$ for the left-hand side to properly evaluate to $\vec{c}^*\Delta\vec{x}(T)$, as will become relevant when deriving the adjoint DAE in Section II-C.

Taking the limit as $\vec{p} \to \vec{0}$, $\Delta\vec{x}$ approaches $\mathbf{M}(t)\Delta\vec{p}$, so

$$\vec{m}^*(T) = \vec{c}^*\mathbf{M}(T) = -\int_A^B \vec{z}^*(t)\mathbf{S}(t)dt. \tag{12}$$

### C. Deriving the Adjoint DAE

The adjoint operator $\mathscr{L}^\dagger\{\vec{y}(t)\} = \vec{z}(t)$ for (10) is

$$-\mathbf{C}^*(t)\frac{d}{dt}\vec{z}(t) + \mathbf{G}^*(t)\vec{z}(t) = \vec{y}(t). \tag{13}$$

This can be verified by evaluating both sides of the inner product relation in (8) and demonstrating their equality. The left-hand side evaluates to

$$\begin{aligned}\langle \Delta\vec{x}(t), \vec{y}(t)\rangle = &-\int_A^B \left(\frac{d}{dt}\vec{z}^*(t)\right)\mathbf{C}(t)\Delta\vec{x}(t)dt \\ &+ \int_A^B \vec{z}^*(t)\mathbf{G}(t)\Delta\vec{x}(t)dt.\end{aligned} \tag{14}$$

Likewise, the right-hand side is

$$\begin{aligned}\langle \vec{u}(t), \vec{z}(t)\rangle = &\int_A^B \vec{z}^*(t)\left(\frac{d}{dt}\mathbf{C}(t)\Delta\vec{x}(t)\right)dt \\ &+ \int_A^B \vec{z}^*(t)\mathbf{G}(t)\Delta\vec{x}(t)dt.\end{aligned} \tag{15}$$

Applying integration by parts, it can be shown that (15) is equivalent to (14), plus a boundary term of

$$[\vec{z}^*(t)\mathbf{C}(t)\Delta\vec{x}(t)]\Big|_A^B. \tag{16}$$

For (13) to be a valid adjoint of (10), the boundary term must evaluate to 0. The initial condition of the original DAE is assumed to not depend on the parameters, so $\Delta\vec{x}(0) = \vec{0}$. If we solve (13) backwards in time from $B$ to $A$, we can choose the final condition to be $\vec{z}(B) = \vec{0}$. Thus, by choosing $A = 0$ and $\vec{z}(B) = \vec{0}$, we can ensure that the boundary term is 0.

Not only does solving the adjoint DAE backwards help us eliminate the boundary term, it is also necessary for maintaining dynamical stability of the solution $\vec{z}(t)$. The original DAE is typically stable in the forward direction, which implies that its adjoint is dynamically unstable in the forward direction.

### D. Deriving Impulsive and Finite Components of the ASF

The input to the adjoint DAE, $\vec{y}(t) = \vec{c}\delta(t-T)$ is infinite-valued and thus not numerically well-defined. So, we cannot solve for the ASF $\vec{z}(t)$ directly through transient analysis. Instead, we derive an expression for $\vec{z}(t)$ of the form

$$\vec{z}(t) = \vec{z}_1(t) + \vec{k}\delta(t-T), \tag{17}$$

where $\vec{z}_1(t)$ is well-defined $\forall t$. In this section, we will determine $\vec{k}$ and $\vec{z}_1(t)$, via integration over an infinitesimal region in $t$. Then, we will demonstrate that the expression for $\vec{m}(T)$ in (12) is numerically well-defined.

First, substitute (17) for $\vec{z}(t)$ into the adjoint DAE and integrate both sides from $T^-$ to $T^+$.

$$\begin{aligned}&-\int_{T^-}^{T^+} \mathbf{C}^*(t)\frac{d}{dt}\left(\vec{z}_1(t) + \vec{k}\delta(t-T)\right)dt \\ &+\int_{T^-}^{T^+} \mathbf{G}^*(t)\left(\vec{z}_1(t) + \vec{k}\delta(t-T)\right)dt = \int_{T^-}^{T^+} \vec{c}\delta(t-T)dt.\end{aligned} \tag{18}$$

By applying integration by parts to the first integral and utilizing the sifting property of the Dirac $\delta$, it can be shown that (18) simplifies to

$$\mathbf{C}^*(T)\vec{z}_1(T^-) + \left(\frac{d}{dt}\mathbf{C}^*(T) + \mathbf{G}^*(T)\right)\vec{k} = \vec{c}. \tag{19}$$

The above equation allows us to find values of $\vec{k}$ and $\vec{z}_1(T^-)$. Noting that, for a DAE with algebraic components, $\mathbf{C}(T)$ will have a nontrivial null space, split $\vec{c}$ into two components: one in the column space of $\mathbf{C}^*(T)$, and one in the null space of $\mathbf{C}(T)$. Since these two subspaces are orthogonal complements of each other, we can rewrite $\vec{c}$ as

$$\vec{c} = \text{proj}_{\text{Col}(\mathbf{C}^*(T))}\vec{c} + \text{proj}_{\text{Null}(\mathbf{C}(T))}\vec{c}. \tag{20}$$

The first term of (19), $\mathbf{C}^*(T)\vec{z}_1(T^-)$, must lie in the column space of $\mathbf{C}^*(T)$, so we set

$$\mathbf{C}^*(T)\vec{z}_1(T^-) = \text{proj}_{\text{Col}(\mathbf{C}^*(T))}\vec{c}. \tag{21}$$

$\vec{z}_1(T^-)$ can be found by solving the above equation. Since the right-hand side is in the column space of $\mathbf{C}^*(T)$, a solution is guaranteed to exist even if $\mathbf{C}^*(T)$ is rank-deficient.

Likewise, we can set the second term of (19) to the projection of $\vec{c}$ onto the null space of $\mathbf{C}(T)$ and find $\vec{k}$ by solving

$$\left(\frac{d}{dt}\mathbf{C}^*(T) + \mathbf{G}^*(T)\right)\vec{k} = \text{proj}_{\text{Null}(\mathbf{C}(T))}\vec{c}. \tag{22}$$

Now, we can find $\vec{z}_1(t)$ by solving the adjoint DAE in (13) backwards from $t = T^-$ to $t = 0$ with final condition $\vec{z}_1(T^-)$. Over $t \in [0, T^-]$, $\vec{z}(t) = \vec{z}_1(t)$ and $\vec{y}(t) = \vec{0}$, so (13) becomes

$$-\mathbf{C}^*(t)\frac{d}{dt}\vec{z}_1(t) + \mathbf{G}^*(t)\vec{z}_1(t) = \vec{0}, \tag{23}$$

which can solved numerically, such as with LMS methods [14].

Now that all the components of $\vec{z}(t)$ have been found, we must show that the integral expression for $\vec{m}(T)$ in (12) is well-defined. Substituting $\vec{z}_1(t) + \vec{k}\delta(t-T)$ for $\vec{z}$ and simplifying,

$$\begin{aligned}\vec{m}^*(T) &= -\int_0^B \left(\vec{z}_1^*(t) + \vec{k}^*\delta(t-T)\right)\mathbf{S}(t)dt \\ &= -\int_0^{T^-} \vec{z}_1^*(t)\mathbf{S}(t)dt - \vec{k}^*\mathbf{S}(T).\end{aligned} \tag{24}$$

All components of the above equation are finite-valued, so we can use numerical integration methods to find $\vec{m}(T)$.

### E. Discontinuity in First Timestep of Computing the ASF

Due to the rank deficiency in $\mathbf{C}^*(t)$, there may be an abrupt jump from the final condition $z_1(T^-)$ to $z_1(T^- - h)$, where $h$ is the transient timestep. For instance, consider a simple, illustrative example:

$$-\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\frac{d}{dt}\vec{z}_1(t) + \begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\vec{z}_1(t) = \vec{0}, \vec{z}_1(T^-) = \begin{bmatrix}1\\ 0\end{bmatrix}. \tag{25}$$

For the purpose of this example, let us examine the first step of a Backward Euler transient simulation:

$$-\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\left(\vec{z}_1(T^- - h) - \begin{bmatrix}1\\ 0\end{bmatrix}\right) - h\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\vec{z}_1(T^- - h) = \vec{0}. \tag{26}$$

It is readily apparent by simplifying the above and solving for $\vec{z}_1(T^- - h)$ that, in the limit as the transient timestep goes to 0, $\vec{z}_1(T^- - h)$ approaches $\begin{bmatrix}1\\ -1\end{bmatrix}$.

In theory, the value of the final condition is correct, but it contributes nothing to the integral for $\vec{m}^*(T)$ in (24), since the discontinuity occurs at an infinitesimal distance from $T$. The contribution to the integral is instead $\lim_{h\to 0}\vec{z}_1(T^- - h)$. We can mitigate these accuracies by taking a very small first timestep for the adjoint DAE, and using the resulting value of $\vec{z}_1(t)$ in lieu of $\vec{z}_1(T)$ when evaluating the integral for $\vec{m}^*(T)$. (24) now integrates a continuous function over a region very close to $[0, T^-]$, and will therefore provide a numerically accurate result.

### F. Full Steps of TADsens and Runtime Analysis

In summary, we can compute the parameter sensitivities $\vec{m}^*(T)$ of $o(T) = \vec{c}^*\vec{x}(T)$ as follows:

1) Solve the DAE in (1) for $\vec{x}_{\text{nom}}(t)$, with nominal parameter values $\vec{p}_{\text{nom}}$ and input $\vec{b}_{\text{nom}}(t)$ from 0 to $T$. If the Jacobian matrices of $\vec{q}$ and $\vec{f}$ with respect to $\vec{x}$ are sparse, this will have runtime complexity of $O(nT)$.[7]
2) While solving the above DAE, compute the Jacobian matrices of $\vec{q}$ and $\vec{f}$ with respect to $\vec{x}$ and $\vec{p}$ enumerated in (3) at every

[7] When solving a DAE using transient methods, we run the Newton-Raphson algorithm at every timestep. So, for a sparse system of size $n$, each iteration of a transient simulation would have runtime complexity $O(n)$.

timestep. Assuming all Jacobians are sparse, this calculation has a runtime complexity of $O((n+n_p)T)$. To improve memory efficiency, these may be computed as needed in subsequent steps.

3) Compute the null space of $\mathbf{C}(T)$ and the column space of $\mathbf{C}^*(T)$, and the projection of $\vec{c}$ onto those two subspaces. If $\mathbf{C}^*(T)$ is a sparse matrix of size $n$, this can be done in $O(n)$ time, such as by LU factorization.
4) Compute $\vec{k}$, the vector scaling factor to the $\delta$-function constituent of $\vec{z}(t)$, as the solution to (22). This (sparse) matrix-vector equation can be solved in $O(n)$ time via LU factorization, or an equivalently efficient method.
5) Compute the final condition $\vec{z}_1(T^-)$ by solving (21). As above, this can be done in $O(n)$ time.
6) Solve adjoint DAE (13) backwards from $T^-$ to 0 for $\vec{z}_1(t)$. This size-$n$ DAE can be solved in $O(nT)$ time.
7) Numerically integrate (24) to get $\vec{m}^*(T)$. Assuming sparse Jacobians, this takes $O((n+n_p)T)$ time.

Overall, we can compute the parameter sensitivities, $\vec{m}^*(T)$, of a DAE output with a runtime and memory complexity of $O((n+n_p)T)$. For DAEs with large numbers of parameters, such as most circuit DAEs, this is significantly better than the $O(nn_pT)$ complexity provided by the direct method.

### G. Insights into Adjoint Parameter Sensitivities for ODEs and Algebraic Equations

We can provide additional insight into the effect of a $\delta$-function input on the solution of the adjoint DAE by examining ODEs and Algebraic Equations. In this section, we use the theory from Section II-D to build intuition behind the separation of $\vec{z}(t)$ into a delta function and a finite transient waveform.

Consider an ODE,

$$\frac{d}{dt}\vec{x}(t)+\vec{f}(\vec{x}(t),\vec{p})+\vec{b}(t)=0. \tag{27}$$

$\mathbf{C}(t)$, the Jacobian matrix of $\vec{q}$ with respect to $\vec{x}$, is the size-$n$ identity matrix, $\mathbf{I_n}$. $\vec{q}$ does not depend on any parameters, so $\mathbf{S_q}$, the Jacobian matrix of $\vec{q}$ with respect to $\vec{p}$, is $\mathbf{0}$.

The final condition of $\vec{z}_1(t)$ is $\vec{z}_1(T^-)=\text{proj}_{\text{Col}(\mathbf{I_n})}\vec{c}$, or $\vec{z}_1(T^-)=\vec{c}$. $\vec{k}$ is the solution to $\left(\frac{d}{dt}\mathbf{I_n}+\mathbf{G}^*(T)\right)\vec{k}=\text{proj}_{\text{Null}(\mathbf{I_n})}\vec{c}$. $\mathbf{I_n}$ has a null space of dimension 0, so $\vec{k}=\vec{0}$. As a result, the adjoint DAE has a finite solution $\forall t$, without a $\delta$-function component despite the $\delta$-function input.

Now, let us consider an algebraic equation,

$$\vec{f}(\vec{x}(t),\vec{p})+\vec{b}(t)=0. \tag{28}$$

This equation does not include a differential component, so $\mathbf{C}(t)$ and $\mathbf{S_q}(t)$ are both $\mathbf{0}$. The null space of $\mathbf{C}(t)$ spans $\mathbb{R}^n$, so $\text{proj}_{\text{Null}(\mathbf{C}(T))}\vec{c}=\vec{c}$ and $\vec{k}$ can be found as $\vec{k}=(\mathbf{G}^*)^{-1}\vec{c}$. The projection of $\vec{c}$ onto the column space of $\mathbf{C}^*(T)$ is $\vec{0}$, making $\vec{z}_1(T^-)=\vec{0}$. Thus, $\vec{z}_1(t)=\vec{0}$, $\forall t\in[0,T^-]$, since it is the output of a linear DAE with a $\vec{0}$ initial condition and an input of $\vec{0}$ over $[0,T^-]$. So, $\vec{z}(t)=\vec{k}\delta(t-T)$.

Intuitively, a system of algebraic equations is purely a function of the current state and inputs, so a $\delta$-function input results solely in a $\delta$-function output.

A DAE can be viewed as a combination of ODEs and algebraic equations, where the ODEs contribute to the column space of $\mathbf{C}^*(t)$, or the linearly independent rows of $\mathbf{C}(t)$, and the algebraic equations contribute the null space of $\mathbf{C}(t)$. The final condition of $\vec{z}_1(t)$, the finite part of the ASF, depends on the column space of $\mathbf{C}^*(T)$ and therefore the ODE components of the original DAE. Likewise, $\vec{k}$, the scaling factor of the delta, depends on the algebraic equations, as it is derived using the null space of $\mathbf{C}(T)$.

## III. Results

In this section, we apply **TADsens** to three examples of varying size. First, we examine a hand-solvable size-two DAE, plotting direct, adjoint, and analytical solutions as a function of time, as well as the relative error of the adjoint method as a function of the transient method and timestep.

Then, using an implementation in MAPP [12, 13], we perform direct and adjoint sensitivity computations on two larger examples: a BJT Schmitt trigger circuit and a 51-stage ring oscillator. The hand-calculable example demonstrates the accuracy of **TADsens** with respect to the analytical solution. The larger analog circuits show that **TADsens** is not only effective for larger systems, but is also more computationally efficient, by a factor of 300, than the direct method.

### A. Hand-Solvable DAE

To verify our implementation against a hand-calculable example, consider the following DAE:

$$\begin{aligned} C\frac{d}{dt}x_1(t)+\frac{x_1(t)-V_{\text{in}}}{R}&=0,\\ x_2(t)-\frac{t}{RC}&=0, \end{aligned} \tag{29}$$

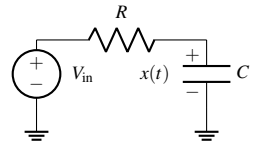


**Fig. 1:** RC circuit with a constant voltage input.

where $x_1(t)$ is the voltage across the capacitor in Figure 1, and $x_2(t)=\frac{t}{RC}$ represents how many RC time constants have passed since the start of simulation.

#### 1) Analytical Solution

We can show that $\vec{x}(t)$ has the solution

$$\begin{bmatrix}x_1(t)\\x_2(t)\end{bmatrix}=\begin{bmatrix}x_1(0)e^{-\frac{t}{RC}}+V_{\text{in}}\left(1-e^{-\frac{t}{RC}}\right)\\ \frac{t}{RC}\end{bmatrix}. \tag{30}$$

The sensitivities of $\vec{x}(t)$ with respect to $\vec{p}=\begin{bmatrix}R\\C\end{bmatrix}$ are

$$\mathbf{M}(t)=\begin{bmatrix}\frac{t}{R^2C}(x_1(0)-V_{\text{in}})e^{-\frac{t}{RC}} & \frac{t}{C^2R}(x_0-V_{\text{in}})e^{-\frac{t}{RC}}\\ -\frac{t}{R^2C} & -\frac{t}{RC^2}\end{bmatrix}. \tag{31}$$

#### 2) Simulation Results

We computed $\vec{m}^*(t)=[2\quad 1]\,\mathbf{M}(t)$ from $t=0$ to $t=10^{-3}$, with $\vec{p}_{\text{nom}}=\begin{bmatrix}1\,k\Omega\\1\,\mu F\end{bmatrix}$, $\vec{x}(0)=\begin{bmatrix}1/2\\0\end{bmatrix}$, and $V_{\text{in}}=1\,V$. In Figure 2, we have plotted the direct, adjoint, and analytically-calculated sensitivities of the system output to $R$, in units of volt per percent deviation from the nominal resistor value.

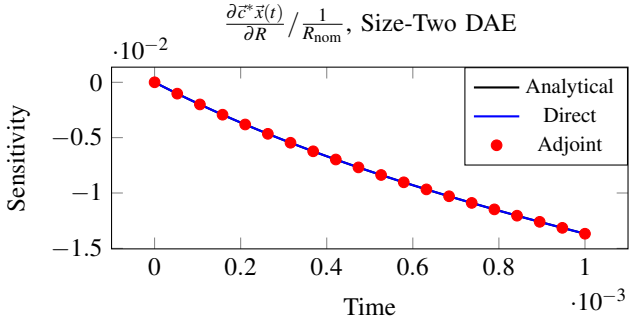


**Fig. 2:** Parameter sensitivity of $\vec{c}^*\vec{x}(t)$ to $R$, in $V/\%$.

We also calculated the average relative error per timestep of the **TADsens**-computed $\vec{m}^*(T=10^{-3})$ to the analytical solution, using transient timesteps ranging from $10^{-6}$ to $10^{-4}$ in Backward Euler, Gear, and Trapezoidal transient simulations. The error is plotted versus transient timestep on a log-log scale in Figure 3. The plot shows that Backward Euler has a quadratic relation between error and timestep, and the other two methods have a cubic relationship. This is consistent with the order of error per timestep expected for each transient simulation [14].

### B. BJT Schmitt Trigger

We now consider the parametric sensitivities of a Schmitt trigger. The Schmitt trigger circuit consists of two BJTs in the configuration pictured in Figure 4. The input voltage is fed into the base of $Q_1$, and there is feedback from the collector node of $Q_1$ to the

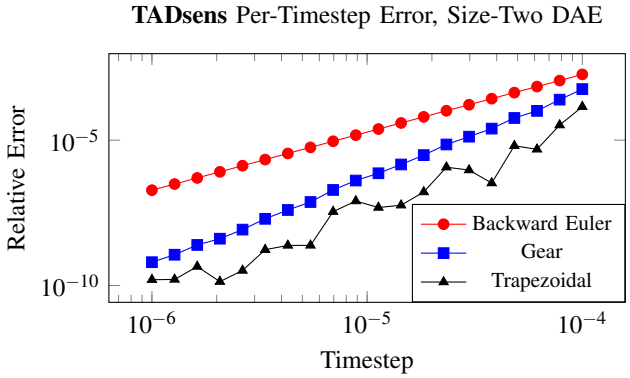


**Fig. 3:** Error of adjoint sensitivities of (29) to analytical solution.

base of $Q_2$. This feedback causes hysteresis to occur, resulting in a digital output signal that does not switch instantaneously if,*e.g.*, noise corrupts the input signal near the switching threshold. The output is measured at the collector node of $Q_2$. This circuit is described by a DAE with four unknowns and 20 parameters, where $Q_1$ and $Q_2$ use the Ebers-Moll BJT model.

A 10kHz pulse signal with a $2V$ amplitude and $0.5V$ DC offset is applied as voltage input. For both direct and adjoint computation, we used second-order Gear transient simulations with a timestep of $2 \cdot 10^{-9}$, over one period of the input waveform.

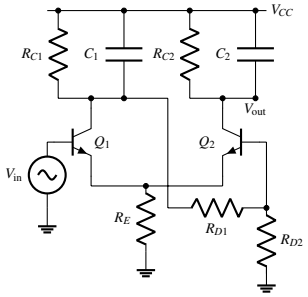


**Fig. 4:** Schmitt trigger circuit.

We then determined the most significant parameters, or those with the largest sensitivities relative to their nominal value. Among the most significant parameters were the supply voltage $V_{CC}$, the forward short-circuit current gain $\alpha_F$ of $Q_2$, and the resistors $R_{C1}$ and $R_{D1}$.

In Figure 5, we have provided tables presenting the parameter sensitivities of the output to these parameters, expressed in volts per percent deviation from $\vec{p}_{\text{nom}}$. In Figure 6, direct and adjoint sensitivities with respect to $\alpha_{F2}$ are plotted over time, with $V_{\text{out}}(t)$ displayed underneath for reference.

**Schmitt Trigger Sensitivities (V/%), Direct Method**

**Computation Time for T=1.00e-04: 121.6 seconds**

| t | $\frac{\partial V_{out}}{\partial V_{CC}} / \frac{1}{V_{CC,nom}}$ | $\frac{\partial V_{out}}{\partial \alpha_{F2}} / \frac{1}{\alpha_{F2,nom}}$ | $\frac{\partial V_{out}}{\partial R_{C1}} / \frac{1}{R_{C1,nom}}$ | $\frac{\partial V_{out}}{\partial R_{D1}} / \frac{1}{R_{D1,nom}}$ |
|---|---|---|---|---|
| 1.613e-05 | 1.02835e-02 | -2.34134e-03 | -2.10356e-03 | 2.12244e-03 |
| 2.662e-05 | -7.01662e-02 | -2.10569e-02 | 3.30722e-02 | -3.31740e-02 |
| 3.711e-05 | 3.52495e-02 | -2.58475e-03 | 4.05963e-03 | -4.07212e-03 |
| 4.760e-05 | 4.81894e-02 | -3.17280e-04 | 4.98322e-04 | -4.99856e-04 |
| 6.857e-05 | 4.99727e-02 | -4.78067e-06 | 7.50857e-06 | -7.53168e-06 |
| 7.906e-05 | 4.99965e-02 | -5.88964e-07 | 1.02232e-06 | -9.58851e-07 |
| 8.955e-05 | -4.51118e-02 | -9.94840e-02 | 6.61450e-02 | -4.59882e-02 |

**Schmitt Trigger Sensitivities (V/%), Adjoint Method**

**Computation Time for T=1.00e-04: 11.1 seconds**

| t | $\frac{\partial V_{out}}{\partial V_{CC}} / \frac{1}{V_{CC,nom}}$ | $\frac{\partial V_{out}}{\partial \alpha_{F2}} / \frac{1}{\alpha_{F2,nom}}$ | $\frac{\partial V_{out}}{\partial R_{C1}} / \frac{1}{R_{C1,nom}}$ | $\frac{\partial V_{out}}{\partial R_{D1}} / \frac{1}{R_{D1,nom}}$ |
|---|---|---|---|---|
| 1.613e-05 | 1.02836e-02 | -2.34142e-03 | -2.10353e-03 | 2.12247e-03 |
| 2.662e-05 | -7.01612e-02 | -2.10562e-02 | 3.30703e-02 | -3.31721e-02 |
| 3.711e-05 | 3.52501e-02 | -2.58466e-03 | 4.05940e-03 | -4.07189e-03 |
| 4.760e-05 | 4.81895e-02 | -3.17269e-04 | 4.98294e-04 | -4.99827e-04 |
| 6.857e-05 | 4.99727e-02 | -4.78051e-06 | 7.50815e-06 | -7.53126e-06 |
| 7.906e-05 | 4.99965e-02 | -5.88944e-07 | 1.02225e-06 | -9.58793e-07 |
| 8.955e-05 | -4.51113e-02 | -9.94839e-02 | 6.61447e-02 | -4.59880e-02 |

**Fig. 5:** Schmitt trigger sensitivities, in volts per percent of $\vec{p}_{\text{nom}}$.

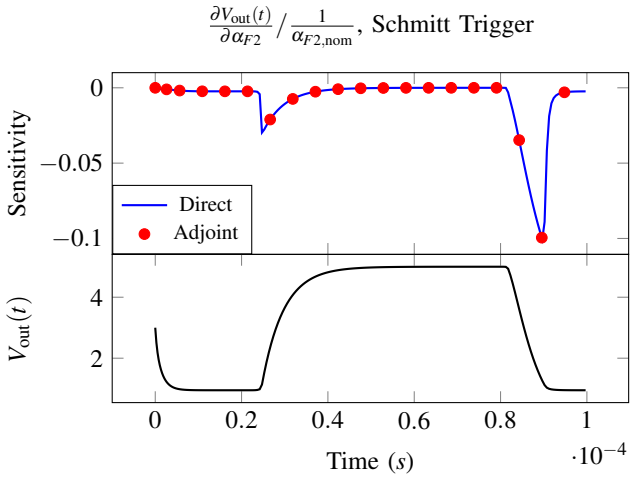


**Fig. 6:** Sensitivity of the Schmitt trigger output to $\alpha_{F2}$. The lower figure shows the nominal output waveform.

We recorded computation times of the direct and adjoint methods using the same timestep, LMS (Linear Multi Step [16]) method, and an end time of $T = 10^{-4}$, but neglecting the time spent computing $x_{\text{nom}}(t)$ and the appropriate Jacobian matrices—this time is common to both direct and adjoint computation. We found that the adjoint method was approximately 10 times faster than the direct method, with a computation of 11.1 seconds, as opposed to 121.6 seconds. This speedup is relatively small due to the size of the circuit: for much larger circuits, such as the ring oscillator in the next section, the adjoint method is seen to be over 100 times faster than direct computation.

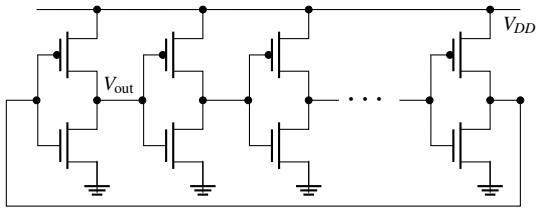


**Fig. 7:** Circuit diagram of ring oscillator

### C. 51-Stage Ring Oscillator

For our final example, we examine a ring oscillator consisting of 51 chained MOS inverters, which is represented by a DAE with 155 unknowns and 664 parameters. We computed the sensitivities of the output of the first inverter over one period of oscillation, and then used these sensitivities to compute the sensitivity of the oscillation period to the parameters.

We first ran a trapezoidal LMS simulation for several periods of oscillation, and found the period to be $2.447 \cdot 10^{-3}$. Inspecting the transient waveform, we found a periodic initial condition, *i.e.*, an initial condition that immediately results in a periodic waveform. We then computed direct and adjoint sensitivities over one oscillator period, using a trapezoidal transient situation with 3000 timepoints.

As the output is switching, the most significant parameters are the load capacitances ($C_L$) at the output of the second, fifty-first, third, and fiftieth inverters. These are the stages feeding and loading the output inverter and are themselves also switching at roughly the same times. In Figure 8, we have displayed the sensitivities of $V_{\text{out}}$ with respect to these parameters, in volts per percent deviation from the nominal capacitance. The sensitivities of $V_{\text{out}}(t)$ to $C_{L2}$ are plotted in Figure 9, with $V_{\text{out}}(t)$ shown below for reference. **TADsens** computed the sensitivities at $T = 2.447 \cdot 10^{-3}$ in 24.6 seconds, 300 times faster than the direct method's 7392 seconds.

We can use the value of $\vec{m}^*(T_{\text{osc, nom}}) = \frac{dV_{\text{out}}(t)}{d\vec{p}}\big|_{t=T_{\text{osc, nom}}}$ we have computed to find the parametric sensitivities of the oscillator period, $T_{\text{osc}}$. We provided the DAE with a periodic initial condition, so

**Ring Oscillator Sensitivities (V/%), Direct Method**

**Computation Time (for T=2.447e-03): 7392 seconds**

| t | $\frac{\partial V_{out}}{\partial C_{L2}} / \frac{1}{C_{L2,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L51}} / \frac{1}{C_{L51,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L4}} / \frac{1}{C_{L4,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L50}} / \frac{1}{C_{L50,nom}}$ |
|---|---|---|---|---|
| 4.910e-04 | -4.18740e-17 | -5.61312e-13 | -2.02375e-23 | -4.56978e-16 |
| 7.364e-04 | -1.29366e-21 | -1.73412e-17 | -6.25219e-28 | -1.41179e-20 |
| 1.227e-03 | 1.72563e-02 | 1.76001e-02 | 1.64867e-02 | 1.63961e-02 |
| 1.473e-03 | 9.49880e-07 | 1.01082e-06 | 9.07513e-07 | 9.04581e-07 |
| 1.718e-03 | 2.93461e-11 | 3.12287e-11 | 2.80372e-11 | 2.79466e-11 |
| 2.209e-03 | 2.80088e-20 | 2.97947e-20 | 2.67598e-20 | 2.66733e-20 |
| 2.447e-03 | -3.80260e-02 | -3.74976e-02 | -3.53447e-02 | -3.51341e-02 |

**Ring Oscillator Sensitivities (V/%), Adjoint Method**

**Computation Time (for T=2.447e-03): 24.6 seconds**

| t | $\frac{\partial V_{out}}{\partial C_{L2}} / \frac{1}{C_{L2,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L51}} / \frac{1}{C_{L51,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L4}} / \frac{1}{C_{L4,nom}}$ | $\frac{\partial V_{out}}{\partial C_{L50}} / \frac{1}{C_{L50,nom}}$ |
|---|---|---|---|---|
| 4.910e-04 | -4.15253e-17 | -5.43021e-13 | -2.00828e-23 | -4.39843e-16 |
| 7.364e-04 | -1.28288e-21 | -1.67761e-17 | -6.20441e-28 | -1.35885e-20 |
| 1.227e-03 | 1.73271e-02 | 1.76544e-02 | 1.65806e-02 | 1.64862e-02 |
| 1.473e-03 | 9.53771e-07 | 1.01257e-06 | 9.12678e-07 | 9.09480e-07 |
| 1.718e-03 | 2.94663e-11 | 3.12830e-11 | 2.81968e-11 | 2.80980e-11 |
| 2.2d09e-03 | 2.81238e-20 | 2.98567e-20 | 2.69121e-20 | 2.68178e-20 |
| 2.447e-03 | -3.82135e-02 | -3.76212e-02 | -3.55460e-02 | -3.53277e-02 |

**Fig. 8:** Ring oscillator sensitivities, in volts per percent of $\vec{p}_{\text{nom}}$.

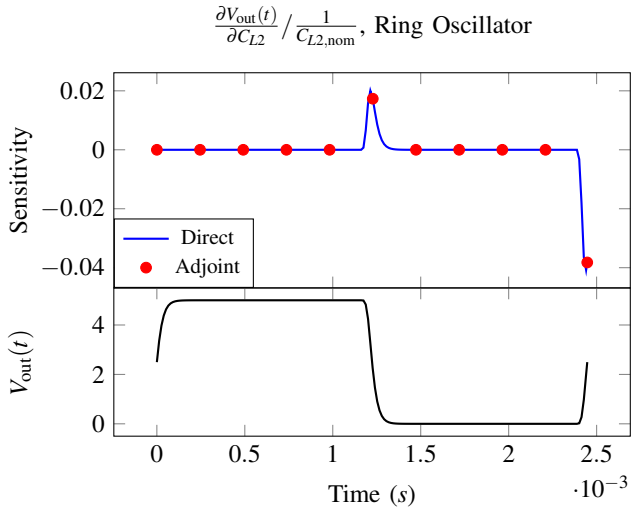


**Fig. 9:** Parameter sensitivity of the output of the first inverter to $C_{L2}$; the lower plot is of the nominal output.

the following must hold:

$$V_{\text{out}}(T_{\text{osc}}, \vec{p}) - V_{\text{out}}(0, \vec{p}) = 0. \tag{32}$$

Taking the derivative of both sides with respect to the parameters and applying the chain rule,

$$\frac{\partial V_{\text{out}}(T_{\text{osc}}, \vec{p})}{\partial \vec{p}} + \frac{\partial V_{\text{out}}(T_{\text{osc}}, \vec{p})}{\partial T_{\text{osc}}} \frac{dT_{\text{osc}}}{d\vec{p}} = 0. \tag{33}$$

So, the parameter sensitivity of the oscillator period can be found by evaluating

$$\frac{dT_{\text{osc}}}{d\vec{p}} = -\frac{\vec{m}^*(T_{\text{osc, nom}})}{\frac{\partial V_{\text{out}}(T_{\text{osc}}, \vec{p})}{\partial T_{\text{osc}}}\Big|_{T_{\text{osc, nom}}, \vec{p}_{\text{nom}}}}. \tag{34}$$

We found the time derivative of $V_{\text{out}}$ at $T_{\text{osc}}$, for nominal parameters, to be $7.477 \cdot 10^4$ V/s. Using this information, we calculated the sensitivity of the oscillator period with respect to the four most significant parameters, as shown in Figure 10.

**Sensitivity of Oscillator Period (% of $T_{\text{osc, nom}}$/% of $\vec{p}_{\text{nom}}$)**

| | $\frac{\partial T_{osc}}{\partial C_{L2}} / \frac{T_{osc,nom}}{C_{L2,nom}}$ | $\frac{\partial T_{osc}}{\partial C_{L51}} / \frac{T_{osc,nom}}{C_{L51,nom}}$ | $\frac{\partial T_{osc}}{\partial C_{L4}} / \frac{T_{osc,nom}}{C_{L4,nom}}$ | $\frac{\partial T_{osc}}{\partial C_{L50}} / \frac{T_{osc,nom}}{C_{L50,nom}}$ |
|---|---|---|---|---|
| **Direct** | 2.0784e-02 | 2.0495e-02 | 1.9318e-02 | 1.9203e-02 |
| **Adjoint** | 2.0886e-02 | 2.0562e-02 | 1.9428e-02 | 1.9309e-02 |

**Fig. 10:** Sensitivities of ring oscillator period, in percent of nominal period per percent deviation from $\vec{p}_{\text{nom}}$.

## IV. Conclusion

We have presented **TADsens**, a theoretically rigorous, numerically accurate, and computationally efficient formulation for transient adjoint parameter sensitivities of a DAE output. We have developed theory for applying adjoints to the DAE sensitivity problem, yielding concrete insights into the response of the adjoint DAE to a $\delta$-function input; in particular, showing that the response consists of $\delta$-function and finite components. We have demonstrated that both components can be computed in a numerically well-posed manner. We have implemented **TADsens** in Berkeley MAPP, a MATLAB® simulator freely available on github. Through three circuit examples, we have confirmed that **TADsens** closely matches results obtained analytically and from direct sensitivity computation. The latter two examples also demonstrate that **TADsens** is viable for larger circuits with high degrees of nonlinearity. We have shown that, for large circuits, **TADsens** is more than two orders of magnitude faster than direct computation.

## Acknowledgments

JR thanks Eric Keiter for bringing the problem to his attention. Support from Sandia National Laboratory and the U.S. National Science Foundation (NSF) is gratefully acknowledged.

## References

[1] U. M. Ascher and L. R. Petzold, *Computer methods for ordinary differential equations and differential-algebraic equations*. Philadelphia: SIAM, 1998.

[2] J. Roychowdhury, *Numerical Simulation and Modelling of Electronic and Biochemical Systems*, vol. 3. NOW Publishers, December 2009.

[3] G. Strang, *Introduction to Linear Algebra*. Wellesley-Cambridge Press, Wellesley, MA, 1993.

[4] S. Director and R. Rohrer, "The Generalized Adjoint Network and Network Sensitivities," *IEEE Trans. Ckt. Theory*, vol. 16, pp. 318–323, August 1969.

[5] D. Hocevar, P. Yang, T. Trick, and B. Epler, "Transient Sensitivity Computation for MOSFET Circuits," *IEEE Trans. CAD*, vol. 4, pp. 609–620, October 1985.

[6] P. Feldmann, T. V. Nguyen, S. W. Director, and R. A. Rohrer, "Sensitivity computation in piecewise approximate circuit simulation," *IEEE Trans. CAD*, vol. 10, no. 2, pp. 171–183, 1991.

[7] F. Liu and P. Feldmann, "A time-unrolling method to compute sensitivity of dynamic systems," in *Proc. IEEE DAC*, June 2014.

[8] K.V. Aadithya, E. Keiter and T. Mei, "DAGSENS: Directed acyclic graph based direct and adjoint transient sensitivity analysis for event-driven objective functions," in *Proc. ICCAD*, pp. 155–162, Nov. 2017.

[9] Y. Cao, S. Li and L. Petzold, "Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software," *Journal of Computational and Applied Mathematics*, vol. 149, pp. 171–191, December 2002.

[10] Y. Cao, S. Li, L. Petzold, and R. Serban, "Adjoint sensitivity analysis for differential-algebraic equations: The adjoint DAE system and its numerical solution," *SIAM Journal on Scientific Computing*, vol. 24, no. 3, pp. 1076–1089, 2003.

[11] A. Meir and J. Roychowdhury, "BLAST: Efficient Computation of Nonlinear Delay Sensitivities in Electronic and Biological Networks using Barycentric Lagrange enabled Transient Adjoint Analysis," in *Proc. IEEE DAC*, June 2012.

[12] T. Wang, A.V. Karthik, B. Wu, J. Yao, and J. Roychowdhury, "MAPP: The Berkeley Model and Algorithm Prototyping Platform," in *Proc. IEEE CICC*, pp. 461–464, September 2015.

[13] "Berkeley MAPP source code." Downloadable from web site https://github.com/jaijeet/MAPP.

[14] L. Chua and P.-M. Lin, *Computer-aided analysis of electronic circuits: algorithms and computational techniques*. Englewood Cliffs, N.J.: Prentice-Hall, 1975.

[15] A. Oppenheim and A. Willsky, *Signals and Systems*. Prentice-Hall, 1997.

[16] C. Gear, *Numerical initial value problems in ordinary differential equations*. Prentice-Hall series in automatic computation, Englewood Cliffs, N.J.: Prentice-Hall, 1971.