UNIVERSITY OF CALIFORNIA SAN DIEGO

Advanced Integration Algorithms for VLSI Circuit Transient Simulation

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Computer Engineering)

by

Xinyuan Wang

Committee in charge:

Professor Chung-Kuan Cheng, Chair
Professor Li-Tien Cheng
Professor Melvin Leok
Professor Vitaliy Lomakin
Professor Yuan Taur

2020

The dissertation of Xinyuan Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Chair

University of California San Diego

2020

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Foremost, I would like to thank my advisor, Professor Chung-Kuan Cheng, for his continuous support and research guidance. He provided me the opportunity to explore the circuit transient simulation algorithms from the perspective of mathematics as well as real applications. During my Ph.D. period, I have learned how to address a challenging problem with positive attitude, perseverance, and passion. In addition to being a very knowledgeable professor, his is considerate for all the students. I would like to express my special thank to Professor Pengwen Chen. He provided brilliant advice in our research collaboration, and helped me with kindness and patience on the research. Thank Professors Li-Tien Cheng, Melvin Leok, Vitaliy Lomakin and Yuan Taur, who serve as my Ph.D. committee members. I appreciate your advice for my research.

Thank all my colleagues, collaborators and friends at UC San Diego, including but not limited to Keming Cao, Kunyao Chen, Xirui Chen, Sijie Ding, Chia-Tung Ho, Chester Holtz, Ilgweon Kang, Daeyeal Lee, Ting-Chou Lin, Dongwon Park, Le Wang, Lutong Wang, Jun Wang, Wenchuan Wei, Yen-Yi Wu, Yu Xin, Bentao Zhang and Hao Zhuang. Especially thank Hao for the guidance on the simulation research. Thank Ilgweon for maintaining the lab servers and sharing the research and life experience. Thank Dongwon for research collaborations. In addition, I would like to thank my best friends, Gao Deng and Jiaqi Mu, for their understanding and support during my Ph.D. period.

And in particular, I want to thank my parents, Binfang Wang and Juhong Yu, for their continuous, selfless love and support.

This work would not have been possible without the support from National Science Foundation (NSF, CCF-1564302).

Chapter 2 is a reprint of the material in the work: H. Zhuang, X. Wang, Q. Chen, P. Chen, and C. K. Cheng, "From circuit theory, simulation to $spice^{Diego}$: A matrix exponential approach for time-domain analysis of large-scale circuits," *IEEE Circuits and*

Systems Magazine, 16(2):16–34, 2016. The author is one of the primary authors and investigators of this work.

Chapter 3 is a combination of the material in the following two works: P. Chen, C. K. Cheng, D. Park, and X. Wang, "Transient circuit simulation for differential algebraic systems using matrix exponential," in *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018. X. Wang, P. Chen, and C.-K. Cheng, "Stability and convergency exploration of matrix exponential integration on power delivery network transient simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019. The author is the primary author and investigator of the papers. The chapter also contains the content the work: P. Chen, C.-K. Cheng, and X. Wang, "Arnoldi algorithms with structured orthogonalization," *arXiv preprint arXiv:2005.14468*, 2020. The author is one of the primary authors and investigators of this work.

Chapter 4 is a reprint of the material as it appears in the following two works: P. Chen, C. K. Cheng, D. Park, and X. Wang, "Transient circuit simulation for differential algebraic systems using matrix exponential," in *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018. X. Wang, P. Chen, and C.-K. Cheng, "Stability and convergency exploration of matrix exponential integration on power delivery network transient simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019. The author is the primary author and investigator of the papers.

Chapter 5, in part, is a reprint of the material in the work: X. Wang, H. Zhuang, and C. K. Cheng, "Exploring the exponential integrators with krylov subspace algorithms for nonlinear circuit simulation," in *Proceedings of the 36th International Conference on Computer-Aided Design*, pages 163–168. IEEE Press, 2017. The author is the primary author and investigator of this work. The chapter also contains the submission for publication of the material in the work: C.-K. Cheng, C.-T. Ho, C. Jiao, X. Wang, Z. Zeng, and X.

Zhan, "A parallel-in-time circuit simulator for power delivery networks with nonlinear load models," submitted to *the 29th Conference on Electrical Performance of Electronic Packaging and Systems*, 2020. The author is one of the primary authors and investigators of this work.

VITA

2014	B. S. in Microelectronics, Tsinghua University
2020	Ph. D. in Electrical Engineering (Computer Engineering), University of California San Diego

PUBLICATIONS

H. Zhuang, X. Wang, Q. Chen, P. Chen, and C. K. Cheng. From circuit theory, simulation to *spice^{Diego}*: A matrix exponential approach for time-domain analysis of large-scale circuits. *IEEE Circuits and Systems Magazine*, 16(2):16–34, 2016.

X. Wang, H. Zhuang, and C. K. Cheng. Exploring the exponential integrators with krylov subspace algorithms for nonlinear circuit simulation. In *Proceedings of the 36th International Conference on Computer-Aided Design*, pages 163–168. IEEE Press, 2017.

P. Chen, C. K. Cheng, D. Park, and X. Wang. Transient circuit simulation for differential algebraic systems using matrix exponential. In *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018.

X. Wang, P. Chen, and C.-K. Cheng. Stability and convergency exploration of matrix exponential integration on power delivery network transient simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

P. Chen, C.-K. Cheng, and X. Wang. Arnoldi algorithms with structured orthogonalization. *arXiv preprint arXiv:2005.14468*, 2020.

C.-K. Cheng, C.-T. Ho, C. Jiao, X. Wang, Z. Zeng, and X. Zhan. A parallel-in-time circuit simulator for power delivery networks with nonlinear load models. Submitted to *the 29th Conference on Electrical Performance of Electronic Packaging and Systems*, 2020.

ABSTRACT OF THE DISSERTATION

Advanced Integration Algorithms for VLSI Circuit Transient Simulation

by

Xinyuan Wang

Doctor of Philosophy in Electrical Engineering (Computer Engineering)

University of California, San Diego, 2020

Professor Chung-Kuan Cheng, Chair

Efficient and reliable transient simulators are essential for VLSI designs due to the increasing size and design complexity. In this dissertation, we explore the advanced time integration algorithms for solving the dynamical systems of VLSI circuits. We aim to reveal the numerical performance of our proposed simulation framework with matrix exponential based integration algorithms and illustrate its advantages over the conventional methods.

First, we provide the theoretical background of matrix exponential based integration with application to circuit transient simulations. To evaluate the matrix exponential and vector products efficiently, Krylov subspace methods and Arnoldi algorithms are used, among which the rational Krylov subspace method exhibits faster convergence on stiff

dynamical systems than standard Krylov subspace method.

Second, we illustrate the stability of matrix exponential based integration algorithms when applied to the transient simulation of power delivery networks. We propose a stability guaranteed Arnoldi algorithm with structured orthogonalization to generate the rational Krylov subspace. The algorithm is verified through theoretical proof and simulation experiments on general linear systems.

Third, we improve the accuracy and convergence rate of the proposed simulation framework with multiple techniques. The exponential related φ functions are introduced and fully exploited in the calculation of matrix exponential and vector products. We choose the appropriate φ function to reduce numerical error according to the condition of exponential operator. The ratio used to shift the spectrum of original system is further studied for performance improvements. We integrate the advanced techniques into the simulation framework for general system-level PDNs. The adaptive stepping of matrix exponential based integration is validated while stable and converged results are achieved with high accuracy.

Finally, we explore the speedup of nonlinear circuit transient simulations. For analog designs with higher accuracy requirements, we adopt the explicit matrix exponential integration method with residual compensation for nonlinear convergence. For PDNs with nonlinear load models, a parallel-in-time method is prompted to parallelize the total simulation time and distribute the integration steps into multiple processors.

Chapter 1

Introduction

1.1 Circuit Simulation

Circuit simulation is widely used to in the design flow of VLSI circuits prior to the manufacturing. Circuit simulators use a transistor level description of circuits and perform a relatively accurate result [8, 37]. Given a circuit netlist, a circuit simulation process consist the evaluation of physical device models, formulation of the differential and algebraic equations, and application of techniques to solve the equations. SPICE was the first general-purpose circuit simulator framework developed by L. W. Nagel at UC Berkeley in the early 1970s.

Transient simulation is the key component in SPICE [8, 35–37, 44]. In the simulation process, numerical integration algorithms usually decide the efficiency and accuracy of the simulation results. In modern VLSI, huge circuit sizes with millions of elements, high stiffness of systems, and tighter design margins make the transient simulation extremely challenging. High-speed simulation methods with non-degrading accuracy remain in high demand. There has been a large amount of research to improve the integration algorithms [11, 12, 28, 31, 58, 61, 63, 64].

In this dissertation, we focus on the following problems and explore advanced integration algorithms for the VLSI circuit transient simulations:

- The conventional linear multi-step methods belong to the low order approximation and bounded by the Dahlquist barrier. Implicit linear multi-step methods, such as Backward Euler, Trapezoidal and Gear's schemes, are preferred due to better stability. However, the step size is limited to reduce the local truncation error, which makes the simulation of large and stiff systems very time consuming.

- The numerical solution of ordinary differential systems (ODEs) is well studied. Unfortunately, in general, circuits equations are not ODEs. The stability and accuracy of the integration algorithms on solving the differential and algebraic equations (DAEs) remains to be an open problem.

- Matrix exponential based integration algorithms break the limitations of conventional linear multi-step methods. The Krylov subspace is adopted to solve the matrix exponential. The performance of the exponential integration method when applied to the transient simulation of ill-conditioned systems remains to be explored.

- The transient simulation of nonlinear circuit is more difficult to solve. The linearization of nonlinear elements and widely used Newton-Raphson iterations cause extra computations for converged solution. A small enough step is usually required to ensure the convergence.

1.2 Outline

This dissertation focuses on the matrix exponential based time integration method and explores its application to circuit transient simulations on ill-conditioned systems with huge problem size or high stiffness. The matrix exponential based integration overcomes

the drawbacks of the conventional low order multi-step approaches. The matrix exponential solvers have remained to be an interesting topic in decades, which were classified into 19 dubious ways by Moler and Van Loan in 1978 [32]. To best of our knowledge, Saad [47] was the first to provide the theoretical foundation to solve the matrix exponential with Krylov subspace, which was clarified as the twentieth approach in [33]. Many works are reported related to the applications of matrix exponential computed with Krylov subspace approach [2, 22, 23, 40, 49]. The contributions of this thesis are summarized as follows:

- We illustrate the privilege of using matrix exponential based integration method compared to the conventional methods in the circuit transient simulations.
- We investigate the stability of the exponential integration for power network analysis. We propose a new algorithm to generate stable rational Krylov subspaces for the computation of matrix exponential and vector products (MEVPs). We validate the algorithm on linear ill-conditioned systems.
- We improve the evaluation of MEVPs by applying different exponential related formulations and selecting optimal parameters in the construction of rational Krylov subspace to confine the spectrum of original system.
- We integrate the advanced techniques in the total simulation framework and applying the approaches to the system-level PDN transient analysis.
- We explore novel integration methods for the transient simulation of nonlinear systems. We perform experiments on analog designs and PDNs with nonlinear load models for different scopes.

In Chapter 2, we introduce the dynamical system from circuits and the conventional numerical time integration algorithms. We present the matrix exponential based integration algorithm which breaks the limitation of the Dahlquist barrier. In order to accelerate

the computation of MEVPs, different preconditioning methods are applied in the Arnoldi algorithm to construct the Krylov subspace. Performance is compared among the multiple integration methods.

In Chapter 3, we investigate the stability of Krylov subspace used to compute the MEVPs for the PDN transient simulation. A semi-explicit DAE with index one is formulated for the PDNs and used for stability analysis. We propose a modified Arnoldi algorithm with structured orthogonalization to resolve the stability issues caused by the ill-conditioned systems. An error analysis is performed on the evaluation of MEVPs for general RC and RLC networks, showing significant improvements with the modified Arnoldi algorithm.

In Chapter 4, we explore the multiple techniques to further improve the accuracy and convergence rate of our calculations based on the stable results from Chapter 3. The exponential related φ functions are applied to the MEVPs due to numerical accuracy concerns. Knowing the spectrum of the circuit system and current step size in MEVPs, we are able to obtain the results using less dimension of Krylov subspace with appropriate choice of φ functions. Besides, the optimal ratio is exploited to confine the spectrum in rational Krylov subspace. We integrate the devised techniques in the simulation framework and validate the performance on system-level PDNs.

In Chapter 5, we focus on the transient simulation algorithms for nonlinear circuits. We apply the matrix exponential based integration to the transient simulation of analog designs. We propose a residual based compensate iteration for the convergence of results. We also work on the PDNs with nonlinear load models, where the macarocells are characterized with linear elements and current sources at different voltage levels. A parallel-in-time method is adopted to parallelize the sequential time stepping to speedup the whole simulation process.

In Chapter 6, we summarize our contributions and present a future scope in this

area.

Chapter 2

Fundamentals of Circuit Transient Simulation and Numerical Time Integration Algorithms

2.1 Formulation of Circuit Transient Simulation

In order to transfer a circuit to a simulation program (SPICE), one must specify the circuit topology and the element constitutive equations. The circuit topology represents how the circuit elements are connected. The element constitutive equations defines the relations among node voltages and branch currents. Circuit differential equations are enforced by conservation laws, which are usually referred to as the Kirchhoff's current law (KCL) and voltage law (KVL). The circuit components, such as linear resistors, capacitors and inductors, as well as nonlinear devices (MOSFETs), are modeled and stamped into a matrix system via modified nodal analysis (MNA) [21]. The fundamental circuit simulation theory starts from differential equations as follows.

Given the circuit netlist and device models, the general formulation is shown as

follows,

$$\frac{dq(x(t))}{dt} + f(x) = Bu(t), \tag{2.1}$$

where $x(t) \in \mathbb{R}^{n\times 1}$ is the vector of nodal voltages and branch currents and n is the size of unknown variables. The charge/flux is represented by $q \in \mathbb{R}^{n\times 1}$ and $f \in \mathbb{R}^{n\times 1}$ contains the current/voltage terms. The derivative $\frac{dq}{dt}$ represents the energy storage elements, such as capacitors or inductors, which have time-dependent effects. Vector $u(t)$ represents all the external excitations at time t and matrix B inserts the signals to the system. If the element constitutive equations are linearized, we can represent Eq. 2.1 as,

$$\mathcal{C}\dot{x}(t) + \mathcal{G}x(t) = Bu(t) + F(x), \tag{2.2}$$

where matrices $\mathcal{C} \in \mathbb{R}^{n\times n}$ represents the capacitance/inductance elements and $\mathcal{G} \in \mathbb{R}^{n\times n}$ represents the conductance/resistance, respectively. Vector $F(x)$ is the nonlinear dynamics evaluated by device model, which is represented as input to the system. The entries are given by

$$\mathcal{C}_{i,j} = \frac{\partial q_i}{\partial x_j},$$

and

$$\mathcal{G}_{i,j} = \frac{\partial f_i}{\partial x_j},$$

where q_i and f_i represents i-th equation in the system of q and f;

$$x \equiv \begin{pmatrix} x_v \\ x_i \end{pmatrix}, \quad u \equiv \begin{pmatrix} u_i \\ u_v \end{pmatrix}, \quad \mathcal{C} \equiv \begin{pmatrix} C & 0 \\ 0 & L \end{pmatrix}, \quad \mathcal{G} \equiv \begin{pmatrix} G & E \\ -E^T & 0 \end{pmatrix}.$$

where vector x_v represents the node voltages; x_i represents the branch currents; vector u_i is the current input; u_v is the voltage input. The block matrices C, L represent the

capacitance and inductance, respectively. Matrix E is the incident matrix.

Circuit transient simulation involves computing the waveform of $x(t)$ as a function of time. Linear multi-step methods are commonly used to solve the ordinary differential equations (ODE) with initial value. With given initial state $x(t)$ and assumption that the system is unchanged in the step from t to $t+h$, one-step integration methods such as Forward Euler (FE), Backward Euler (BE), Trapezoidal (TR) [37] will be discussed in Chapter 2.2.

2.2 Conventional Numerical Time Integration Algorithms

For simplicity, we start from a linear system as

$$\mathcal{C}\frac{dx}{dt} = -\mathcal{G}x + Bu(t). \tag{2.3}$$

With the initial vector $x(t)$ at time t given, we compute the solution $x(t+h)$ with time step h. Assume that $\mathcal{C}$ is nonsingular, we define the system matrix $A := -\mathcal{C}^{-1}G$.

1. Forward Euler Time Integration (FE): Forward Euler time integration scheme starts with the approximation

$$x(t+h) = x(t) + h\dot{x}(t),$$

which leads to

$$\frac{\mathcal{C}}{h}x(t+h) = \left(\frac{\mathcal{C}}{h} - \mathcal{G}\right)x(t) + Bu(t) \tag{2.4}$$

in the circuit simulation formulation.

2. Backward Euler Time Integration (BE): Backward Euler time integration scheme starts with

$$x(t+h) = x(t) + h\dot{x}(t+h),$$

which gives

$$\left(\frac{\mathcal{C}}{h} + \mathcal{G}\right) x(t+h) = \frac{\mathcal{C}}{h} x(t) + Bu(t+h). \tag{2.5}$$

3. Trapezoidal Time Integration (TR):

$$x(t+h) = x(t) + \frac{h}{2}(\dot{x}(t) + \dot{x}(t+h)),$$

which gives

$$\left(\frac{\mathcal{C}}{h} + \frac{\mathcal{G}}{2}\right) x(t+h) = \left(\frac{\mathcal{C}}{h} - \frac{\mathcal{G}}{2}\right) x(t) + B\frac{u(t) + u(t+h)}{2}. \tag{2.6}$$

Methods FE, BE, and TR all belong to linear multi-step method, also known as the linear one-step method. A-stable linear multi-step methods are favored in circuit simulation to solve time integration problems, since the numerical error is only caused by local truncation error (LTE) and would not be amplified by the instability of numerical integration itself.

Definition 2.2.1 (A-stability)**.** A linear multi-step method is said to be *A-stable* if its region of absolute stability includes the whole left half-plane[1].

[1]Another equivalent way to interpretation of A-stable: The numerical integration method is A-stable. For the linear system $dx/dt = Ax$ with time step h, the solution $x(t+h)$ obtained by the numerical integration approaches 0, or $x(t+h) \to 0$ when $h \to \infty$ and the real parts of all eigenvalues of A are negative.

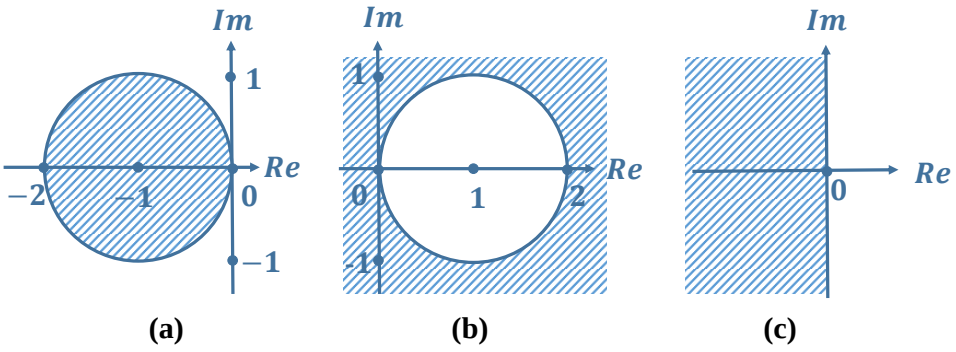

Figure 2.1: Stability regions (shaded) of (a) Forward Euler (FE), (b) Backward Euler (BE), and (c) Trapezoidal methods in the complex plane.

The stability regions of FE, BE and TR are shown in Fig. 2.1. Method FE has a very limited stability region, while BE covers the largest region in the complex plane. Time step h in FE is constrained by $\min(|\lambda_i|^{-1})$ (λ_i: an eigenvalue of matrix A). Electronic circuits have eigenvalue magnitudes spanning at least several decades, which leads to impractically tiny time step h for simulation using FE. Circuit systems with a wide range of eigenvalues are said to be stiff [39]. BE and TR are all A-stable and served as baseline methods in this work. We keep the other linear multi-step schemes out of this paper, since the numerical integration in SPICE-like tools usually use linear multi-step methods so that they cannot exceed the second Dahlquist barrier.

Theorem 2.2.1 (the second Dahlquist barrier)**.** *There are no explicit A-stable and linear multi-step methods. The implicit ones have order of convergence at most* 2*. The trapezoidal rule has the smallest error constant amongst the A-stable linear multi-step methods of order* 2 *[8, 57].*

Interested readers can refer to [8, 37, 44] for more details of numerical stability in circuit simulation.

2.3 A Matrix Exponential Based Integration Algorithm

2.3.1 Time Integration with Matrix Exponential

We start from the linear circuit ODE in Eq. 2.3 which consists a nonsingular $\mathcal{C}$, which could be solved analytically with initial value $x(t)$ [8].

$$x(t+h) = e^{h\mathcal{A}}x(t) + \int_0^h e^{(h-\tau)\mathcal{A}}b(t+\tau)d\tau, \tag{2.7}$$

where $\mathcal{A} = -\mathcal{C}^{-1}\mathcal{G}$ is the system matrix in MEVPs and $b(t) = \mathcal{C}^{-1}Bu(t)$ is the excitation from input sources.

Suppose that the input $u(t)$ is piece-wise-linear (PWL), the solution in Eq. 2.7 can be derived as an explicit expression with matrix exponential and vector products (MEVPs)

$$\begin{aligned} x(t+h) &= x(t) \\ &+ (e^{h\mathcal{A}} - I)\mathcal{A}^{-1}g(t) \\ &+ (e^{h\mathcal{A}} - h\mathcal{A} - I)\mathcal{A}^{-2}\frac{b(t+h) - b(t)}{h}, \end{aligned} \tag{2.8}$$

where the vectors are defined as

$$g(t) = \mathcal{A}x(t) + b(t), \tag{2.9}$$

and the slope of PWL input

$$\frac{db(t)}{dt} = \frac{b(t+h) - b(t)}{h}. \tag{2.10}$$

To best of our knowledge, all of the numerical integration methods in SPICE-like

simulators are from the linear multi-step scheme, which try to approximate the exact solution via matrix exponential operators [8] in a low order way. To simplify the calculation, we consider the homogeneous system with $u(t) = 0$,

$$\frac{dx}{dt} = \mathcal{A}x, \tag{2.11}$$

with solution

$$\begin{aligned} x(t+h) &= e^{h\mathcal{A}}x(t) = \sum_{k=0}^{\infty} \frac{h^k \mathcal{A}^k}{k!} x(t) \\ &= x(t) + h\mathcal{A}x(t) + \frac{h^2\mathcal{A}^2}{2}x(t) + \frac{h^3\mathcal{A}^3}{3!}x(t) + \cdots + \frac{h^k\mathcal{A}^k}{k!}x(t) + \cdots . \end{aligned} \tag{2.12}$$

The conventional linear one-step integration methods attempt a low order polynomial approximation of Eq. 2.12.

1. FE method:

$$x(t+h) = \left(\frac{\mathcal{C}}{h}\right)^{-1} \left(\frac{\mathcal{C}}{h} - \mathcal{G}\right) x(t) = (I + h\mathcal{A})x(t) \tag{2.13}$$

   The formulation fits the first two terms of the exact solution. The accuracy order of FE is $O(h)$.

2. BE method:

$$x(t+h) = \left(\frac{\mathcal{C}}{h} + \mathcal{G}\right)^{-1} \frac{\mathcal{C}}{h} x(t) = (I - h\mathcal{A})^{-1}x(t) \tag{2.14}$$

   The formulation also fits the first two terms of the exact solution. The accuracy order of BE is $O(h)$.

3. TR method:

$$x(t+h) = \left(\frac{\mathcal{C}}{h} + \frac{\mathcal{G}}{2}\right)^{-1}\left(\frac{\mathcal{C}}{h} - \frac{\mathcal{G}}{2}\right)x(t) = (I - \frac{h\mathcal{A}}{2})^{-1}(I + \frac{h\mathcal{A}}{2})x(t) \qquad (2.15)$$

The formulation fits the first three terms of the exact solution. The accuracy order of TR is $O(h^2)$.

Note that the expressions only converge for $h\mathcal{A}$ of BE and $\frac{h\mathcal{A}}{2}$ of TR with spectral radius less than one. Besides, the missing higher order terms introduce the LTE to the conventional methods, which constrain the time step with respect to the region of Taylor expansion.

Fig. 2.2 shows a test equation with size $n = 1$, and is solved directly by function EXPM as well as FE, BE, and TR. The figure illustrates that mismatched results of FE, BE, and TR compared to EXPM with different time step h. In other words, if e^{hA} is used to compute the solution of differential equation system directly, there is no local truncation error constraint for the time step choice. However, the question is how matrix exponential and vector product (MEVP) can be computed in an efficient way, since the size of system could be extremely large which makes the direct computation infeasible. In addition, Fig. 2.3 describes a "hump" effect during the computation of the exponential function [33]. Term $\mathcal{A}^k/k!$ of series in Eq. 2.12 may increase before the value can drop after $k > |\lambda(\mathcal{A})|_{max}$. Therefore, we need a high order k to converge the series, which makes MEVP computation even more challenging.

2.3.2 Evaluation of the Matrix Exponential and Vector Products via Krylov subspace

One efficient way among different approaches is to compute the MEVPs through Krylov subspace method [33, 47]. The complexity of evaluating the MEVPs can be reduced

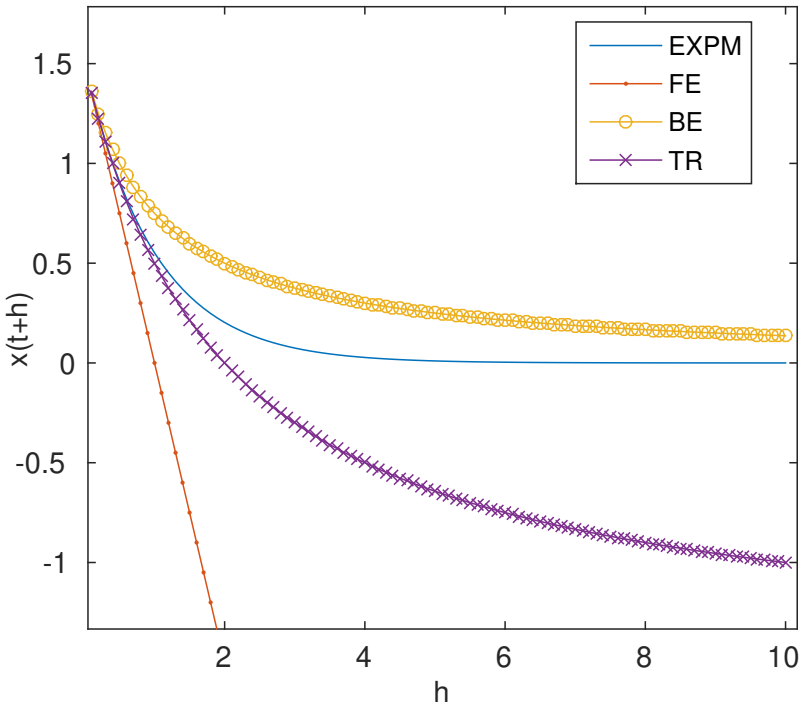

Figure 2.2: A test equation $\frac{dx}{dt} = -x(t)$, where $x(0) = 1.5$, $h \in [0, 10]$. Analytical solution is computed by EXPM $x(h) = e^{-h}x(0)$.

while maintaining a high order polynomial approximation [47].

Definition 2.3.1 (Krylov Subspace)**.** Given a matrix A and a vector v, the Krylov subspace of order m, denoted by $K_m(A, v)$, is defined as the subspace spanned by the vectors $v, Av, \cdots, .A^{m-1}v$, or

$$K_m(A, v) := \text{span}\{v, Av, \cdots, A^{m-1}v\}. \tag{2.16}$$

It is convenient to work with an orthonormal basis for $K_m := K_m(A, v)$. Let $\{v_i\}_{i=1}^m$ be an orthonormal basis for K_m. Let V_m be the $n \times m$ matrix with $\{v_i\}_{i=1}^m$ as its columns. $V_m V_m^\top$ is the projection onto K_m. Let H_m be the $m \times m$ Hessenberg matrix expressing A as an operator restricted to K_m on the basis $\{v_i\}_{i=0}^{m-1}$, i.e., $H_m = V_m^\top A V_m$. We have v,

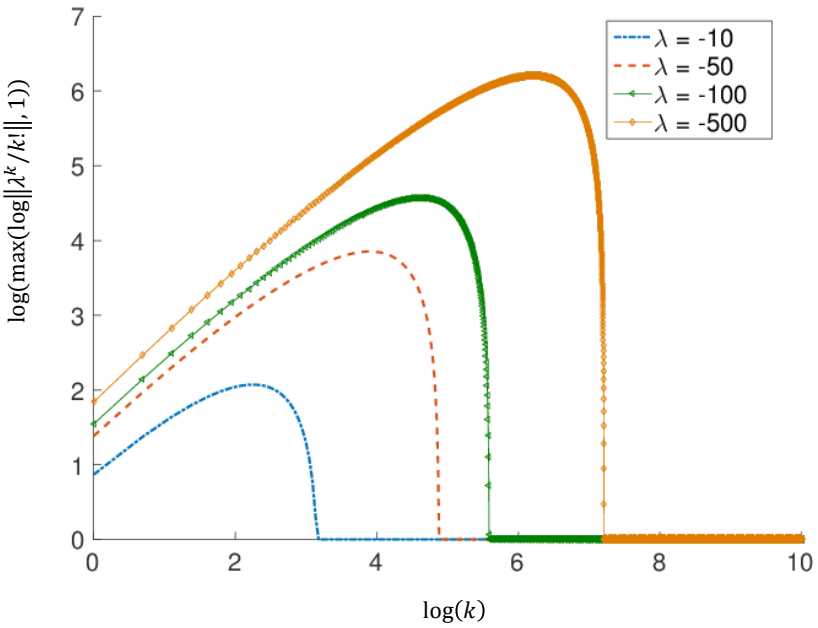

Figure 2.3: The "hump" effect of terms in the Taylor series of the exponential function in [33].

$Av \in K_m$, then

$$\begin{aligned} Av &= (V_m V_m^\top) A (V_m V_m^\top) v \\ &= V_m (V_m^\top A V_m) V_m^\top v \\ &= V_m H_m V_m^T v. \end{aligned} \tag{2.17}$$

Similarly, for all $i \leq m-1$,

$$A^i v = V_m H_m^i V_m^\top v,$$

we have $p(A)v = V_m p(H_m) V_m^T v$, for any polynomial p of degree at most $m-1$ [47].

Lemma 2.3.1 (Exact Computation with Polynomials. See e.g., [43, 47]). Let V_m and H_m be as defined above. For any polynomial p of degree at most $m-1$,

$$p(A)v = V_m p(H_m) V_m^T v. \tag{2.18}$$

Thus, H_m can be used to compute the function $p(A)v$ for any degree $m-1$ polynomial. The Lemma 2.3.1 suggests that a candidate for computing $f(A)v$ approximately is via $V_m f(H_m)V_m^\top v$. The metric to evaluate the result is the norm of error, such as $\|f(A)v - V_m f(H_m)V_m^\top v\|$ [43]. Let p_{m-1} be any polynomial of degree $\leq m-1$ approximating $f(z)$, and define the remainder $r_m(z) = f(z) - p_{m-1}(z)$. Then

$$f(A)v - V_m f(H_m)V_m^\top v = r_m(A)v - V_m r_m(H_m)V_m^\top v. \tag{2.19}$$

Therefore, the error is bounded by the value of r_m on the eigenvalues of A and H_m. For details see [47].

Lemma 2.3.2 (Approximation by Best Polynomial. See e.g., [43,47]). Let V_m and H_m be as defined above. Let $f : \mathbb{R} \to \mathbb{R}$ be any function such that $f(A)$ and $f(H_m)$ are well-defined. Then,

$$\begin{aligned} & \|f(A)v - V_m f(H_m)V_m^\top v\| \\ \leq & \min_{p_{m-1}\in\Sigma_{m-1}} (\max_{\lambda\in\Lambda(A)} |f(\lambda) - p_{m-1}(\lambda)| \\ + & \max_{\lambda\in\Lambda(H_m)} |f(\lambda) - p_{m-1}(\lambda)|). \end{aligned} \tag{2.20}$$

Hence, $V_m f(H_m)V_m^\top v$ approximates $f(A)v$ as well as the best degree $m-1$ polynomial that uniformly approximates f. Arnoldi algorithm (Algorithm 1) is used to construct the orthonormal basis of Krylov subspace starting from the normalized initial vector [47, 58].

The steps from line 4 to 7 of Algorithm 1 form a modified Gram-Schmidt process. The process above produces an orthonormal basis $V_m = [v_1, v_2, \ldots, v_m]$ of the Krylov subspace K_m. The $m \times m$ upper Hessenberg matrix H_m consisting of the $h_{i,j}$ from the algorithm follows the relation

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^\top, \tag{2.21}$$

Algorithm 1: Arnoldi Algorithm

Input: A, v
Output: V_m, H_m

$v_1 = v/\|v\|$;
for $j = 1 : m$ **do**
  $w = Av_j$;
  for $i = 1 : j$ **do**
    $h_{i,j} = w^\top v_i$;
    $w = w - h_{i,j}v_i$;
  end
  $h_{j+1,j} = \|w\|$;
  $v_{j+1} = \frac{w}{h_{j+1,j}}$;
  if $r(h, m)$ ***PASS*** *residue check* **then**
    $m = j$;
    break;
  end
end

where e_m is the m-th unit vector with dimension $m \times 1$. Then, the MEVP $f(A)v = e^{hA}v$ is computed via

$$e^{hA}v \approx \beta V_m e^{hH_m} e_1. \tag{2.22}$$

where $\beta = \|v\|$.

For the homogeneous system in Eq. 2.11, the posterior residual-based error term can be used as termination criteria of Algorithm 1 [3].

$$\|r\| = \|\frac{dx}{dt} - Ax\| = \|\beta h_{m+1,m} v_{m+1} e_m^\top e^{hH_m} e_1\|. \tag{2.23}$$

However, in circuit theory, we actually solve the system

$$C\frac{dx}{dt} = -Gx,$$

which defines the residual (error) approximation

$$\|r(m,h)\| = \|C\frac{dx}{dt} + Gx\| = \|\beta h_{m+1,m} C v_{m+1} e_m^\top e^{hH_m} e_1\| \tag{2.24}$$

for our circuit simulation problem. This also leads to the solving of

$$Cw = -Gv_j$$

as described in line 3 of Algorithm 1.

Note that Eq. 2.22 distinguishes approximation method from linear multi-step methods, which uses non-linear coefficients generated by e^{hH_m}. *Therefore, the matrix exponential methods break away from linear multi-step methods and thus are not limited by the Dahlquist barrier.*

For the accuracy of approximation of $e^{hA}v$, large dimension of Krylov subspace basis is required, which not only increases the computational complexity but also consumes huge amount of memory. The reason is that the Hessenberg matrix H_m and subspace V_m of standard Krylov subspace method tend to approximate the large magnitude eigenvalues and the corresponding eigenvectors of A [53]. Due to the exponential decay of higher order terms in Taylor expansion, such components are not the crux of circuit system's dynamical behaviors [3, 53].

Dealing with stiff circuits, therefore, needs to gather more vectors into subspace basis and increases the size of H_m to fetch more useful components, which results in both memory overhead and computational complexity into the Krylov subspace generations during time stepping.

To improve the efficiency, we adopt the idea from *spectral transformation* [3,13,41,53] to effectively capture small magnitude eigenvalues and corresponding eigenvectors in A, leading to a fast yet accurate MEVP computation.

2.4 Performance of Conventional and Matrix Exponential Based Integration Algorithms

As described in Chapter 2.3, the MEVPs could be calculated with Krylov subspace generated via Arnoldi algorithm. When applied to circuit transient simulation, standard Krylov subspace method cannot work efficiently to capture the dominant components of circuit systems and causes slow convergence. In this section, we introduce the spectral transformation in the construction of Krylov subspace [3, 13, 41, 53], and compare the numerical performance of multiple Krylov subspace methods as well as the conventional integration methods.

2.4.1 Computation of MEVP via Invert Krylov Subspace

Instead of A, we use A^{-1} as our target matrix to form the Krylov subspace

$$K_m(A^{-1}, v) := \text{span}\{v, A^{-1}v, \cdots, A^{-(m-1)}v\}. \tag{2.25}$$

Intuitively, by inverting A the small magnitude eigenvalues become the large ones in A^{-1}. The resulting H_m is likely to capture these eigenvalues first. Based on the Arnoldi algorithm, the invert Krylov subspace has the relation of matrices

$$A^{-1}V_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^\top. \tag{2.26}$$

The MEVP can be approximated as

$$e^{hA}v \approx \beta V_m e^{hH_m^{-1}} e_1. \tag{2.27}$$

The residual approximation follows Eq. 2.24

$$\|r(m,h)\| = \|\beta h_{m+1,m} G v_{m+1} e_m^\top H_m^{-1} e^{hH_m^{-1}} e_1\|. \tag{2.28}$$

In line 3 of Algorithm 1, the generation of new basis vector requires solving

$$Gw = -Cv_j.$$

2.4.2 Computation of MEVP via Rational Krylov Subspace

The shift-and-invert operation [53] is designed to confine the spectrum of A with the parameter γ. The basis of rational Krylov subspace follows

$$K_m((I-\gamma A)^{-1}, v) := \text{span}\{v, (I-\gamma A)^{-1}v, \cdots, (I-\gamma A)^{-(m-1)}v\}. \tag{2.29}$$

With this operation, the magnitude of all eigenvalues is smaller than one. According to [3, 53], the shift-and-invert basis for matrix exponential based transient simulation is not very sensitive to γ, once it is set to around the order near time steps used in transient simulation. The similar idea has been applied to simple power grid simulation with matrix exponential method [65, 66]. The basis vectors V_m and Hessenberg matrix H_m of rational Krylov subspace satisfies the relation

$$(I-\gamma A)^{-1} V_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^\top. \tag{2.30}$$

The MEVP can be approximated as

$$e^{hA} v \approx \beta V_m e^{h\frac{I-H_m^{-1}}{\gamma}} e_1. \tag{2.31}$$

The residual approximation is derived as

$$\|r(m,h)\| = \|\beta h_{m+1,m}(G + \frac{C}{\gamma})v_{m+1}e_m^\top H_m^{-1} e^{h\frac{I-H_m^{-1}}{\gamma}} e_1\|. \tag{2.32}$$

Since we don't directly calculating A^{-1}, the generation of new basis follows

$$(\gamma G + C)w = Cv_j,$$

as described in line 3 of Algorithm 1.

2.4.3 Performance Comparison of Multiple Krylov Subspace Methods

To validate the performance of the standard, invert, and rational Krylov subspace methods, we generate a RC mesh with size $n = 1600$. The entries of conductance in G are in the range $[0.01, 100]$. Each node is connected by a capacitor to ground, which results in a diagonal matrix C with range in $[8.5 \times 10^{-18}, 9.9 \times 10^{-16}]$. Both matrices are nonsingular. The resultant matrix $A = -C^{-1}G$ contains eigenvalues in $[-3.98 \times 10^{17}, -8.49 \times 10^{10}]$ with stiffness

$$\frac{Re(\lambda_{min})}{Re(\lambda_{max})} = \frac{-3.98 \times 10^{17}}{-8.49 \times 10^{10}} = 4.7 \times 10^6,$$

where λ_{max} and λ_{min} are the maximum and minimum eigenvalues of A, respectively.

Start from the initial vector v generated by MATLAB *rand* function and we choose the step size $h = 0.4ps$, the analytical solution of the homogeneous ODE system is calculated using Eq. 2.22, 2.27 and 2.31. The ratio in rational Krylov method is set as $\gamma = 10^{-13}$. The exact solution is calculated with MATLAB *expm* function since the matrices are nonsingular, which serves as the reference solution. Fig. 2.4 shows the relative error reductions along

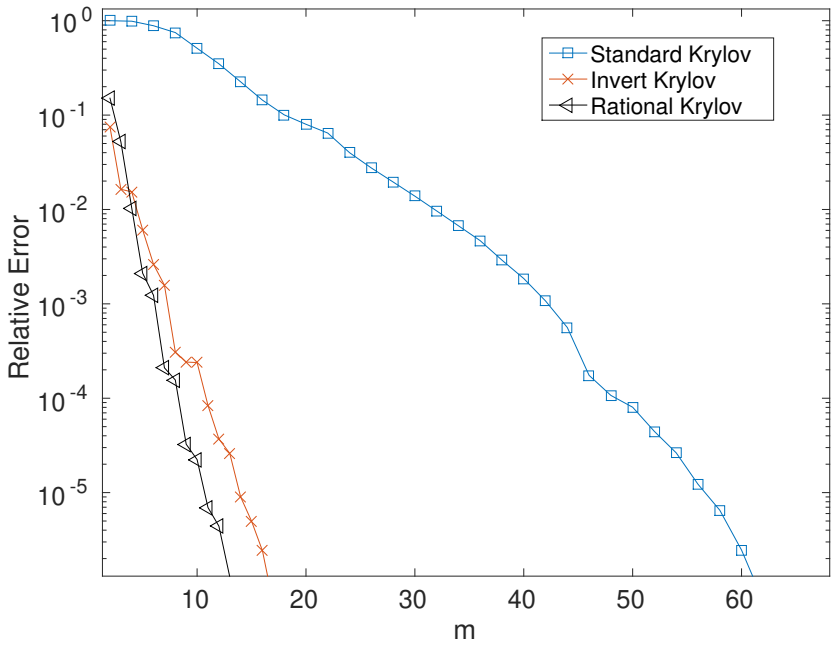

Figure 2.4: The relative error vs. dimensional m of different Krylov subspace methods.

the increasing Krylov subspace dimension. The error reduction rate of rational Krylov subspace is the best, while the one of standard Krylov subspace requires huge dimension to capture the same level of error. For example, it costs almost $10\times$ of the size to achieve around relative error 1% compared to invert and rational Krylov subspace methods.

In order to observe the different stiffness effects on Krylov subspace methods, we modify the values in C and G to increase the stiffness to 4.7×10^{10}. Fig. 2.5 illustrates the stable reduction rate of rational Krylov method. The performance of standard and invert Krylov methods degrades on system with higher stiffness.

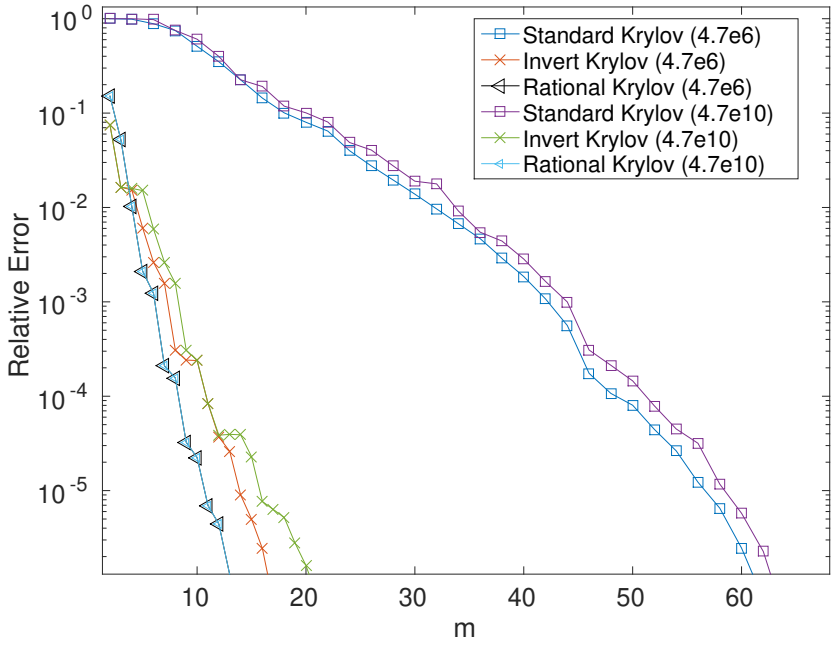

Figure 2.5: The relative error vs. dimension m of different Krylov subspace methods with two stiffness numbers.

2.4.4 Error Distributions of the Numerical Integration Approaches with a Single Time Step

To further explore the numerical performance of the integration methods when applied to circuit transient simulation, the absolute error distribution versus h is plotted in Fig. 2.6 including the standard, invert, and rational Krylov subspace methods as well as FE, BE, and TR methods.

$$Error = \|f(h) - e^{hA}v\|_\infty$$

For rational Krylov subspace, we set $\gamma = h/2$. In most of the cases, a Krylov subspace with higher dimension m could provide more accurate results.

For the case $h \leq min(|\lambda_i|^{-1})$, Taylor expansion is valid for BE and TR. Thus, the

BE method has the error slope following the 2nd order term while TR method following the 3rd order term. In Fig. 2.6(a), there are some abnormal curves of the rational Krylov subspace method (Rat) due to the numerical issues, when h is too small, and the matrix A disappear since $(I - \frac{h}{2}A)^{-1} \rightarrow I$. In Fig. 2.6(b), the curves of standard Krylov method (Std) shift to the right as m increases, while the invert Krylov method (Inv) has opposite trend. Both errors decrease as h becomes smaller until they reach the numerical error floor.

For the case $h \geq max(|\lambda_i|^{-1})$, the solution attenuates globally. Thus, the absolute error curves of Krylov subspace methods drop exponentially. The Inv and Rat methods obtain faster convergence rate compared to Std method. The explanation is that a relatively small portion of the eigenvalues and corresponding invariant subspaces determines the final result (vector) when time step h is larger [53], which are efficiently captured by invert and rational Krylov subspace methods. However, the implicit BE and TR error curves remain flat due to accuracy limitations. The explicit FE method suffers from large LTE.

For the case that h is between the two bounds, most curves drop as the dimension m increases. For this circuit, we are interested in the behavior in the nano-second scale. At this time scale, Inv converges faster than Std as dimension m increases. This summary of error trend is listed in Table 2.1.

2.5 Summary

In this section, we demonstrate the numerical performance of the matrix exponential based integrators. Krylov methods for MEVP can alter their orders to improve accuracy, which is not possible for traditional linear multi-step methods. In general, in a stiff system, simulation can have time step h much larger than the feasible range of Taylor expansion. Traditional linear multi-step approach relies on the marching in time to drive the errors down, while matrix exponential approach can pull down the error by increasing

Table 2.1: Matrix Exponential Based High Order Integrators using Std, Inv, and Rat. vs. Low Order Integrators FE, BE and TR.

Method	$h \leq \min(\lvert\lambda_i\rvert^{-1})$	$\min(\lvert\lambda_i\rvert^{-1}) < h < \max(\lvert\lambda_i\rvert^{-1})$	$h \geq \max(\lvert\lambda_i\rvert^{-1})$
FE	2nd order	Diverge	Diverge
BE	2nd order	Flat	Flat
TR	3rd order	Flat (worse than BE)	Flat
Std ($m = 2$)	2nd order	Flat	Drop
Inv ($m = 2$)	1st order	Flat	Drop
Rat ($m = 2$)	1st order	Flat	Drop
Std ($m > 2$)	>2nd order	Curves shift to the right	Drop
Inv ($m > 2$)	1st order	Curves shift to the left	Drop
Rat ($m > 2$)	*	*	Drop

*: The curve of Rat depends on γ. For large γ, the curve is similar to Inv. For small γ, the curve is similar to Std. Otherwise, the shape of curve falls between Std and Inv. Moreover, for $m = 2$, the curve dips at $h = 2\gamma$. As dimension m increases, the dip point shifts to the right.

the dimension of the Krylov subspace. For transient analysis, the eigenvalues of small real magnitude are wanted to describe the dynamic behavior. Therefore, for the Krylov variants, invert (Inv) and rational (Rat) Krylov methods are good choices.

More importantly, exponential based integration schemes with Krylov subspaces have three distinguished features:

(1) For invert and rational Krylov subspace methods, the larger is time step, the smaller errors we will have. This phenomenon is consistent with the result of van den Eshof and Hochbruck in [53].

(2) Invert Krylov subspace method can avoid the factorization of matrix C, so that it can solve the post-layout simulation when the capacitance/inductance matrix C is complicated (relatively denser than pre-layout, or strong coupled systems), while the complexities by standard methods may increase dramatically.

(3) The explicit formulation is stable by matrix exponential operators and Krylov subspace methods. Thus, for nonlinear system, we can skip the procedures needed in implicit

method such as NR iteration.

Chapter 2 is a reprint of the material in the work: H. Zhuang, X. Wang, Q. Chen, P. Chen, and C. K. Cheng, "From circuit theory, simulation to $spice^{Diego}$: A matrix exponential approach for time-domain analysis of large-scale circuits," *IEEE Circuits and Systems Magazine*, 16(2):16–34, 2016. The author is one of the primary authors and investigators of this work.

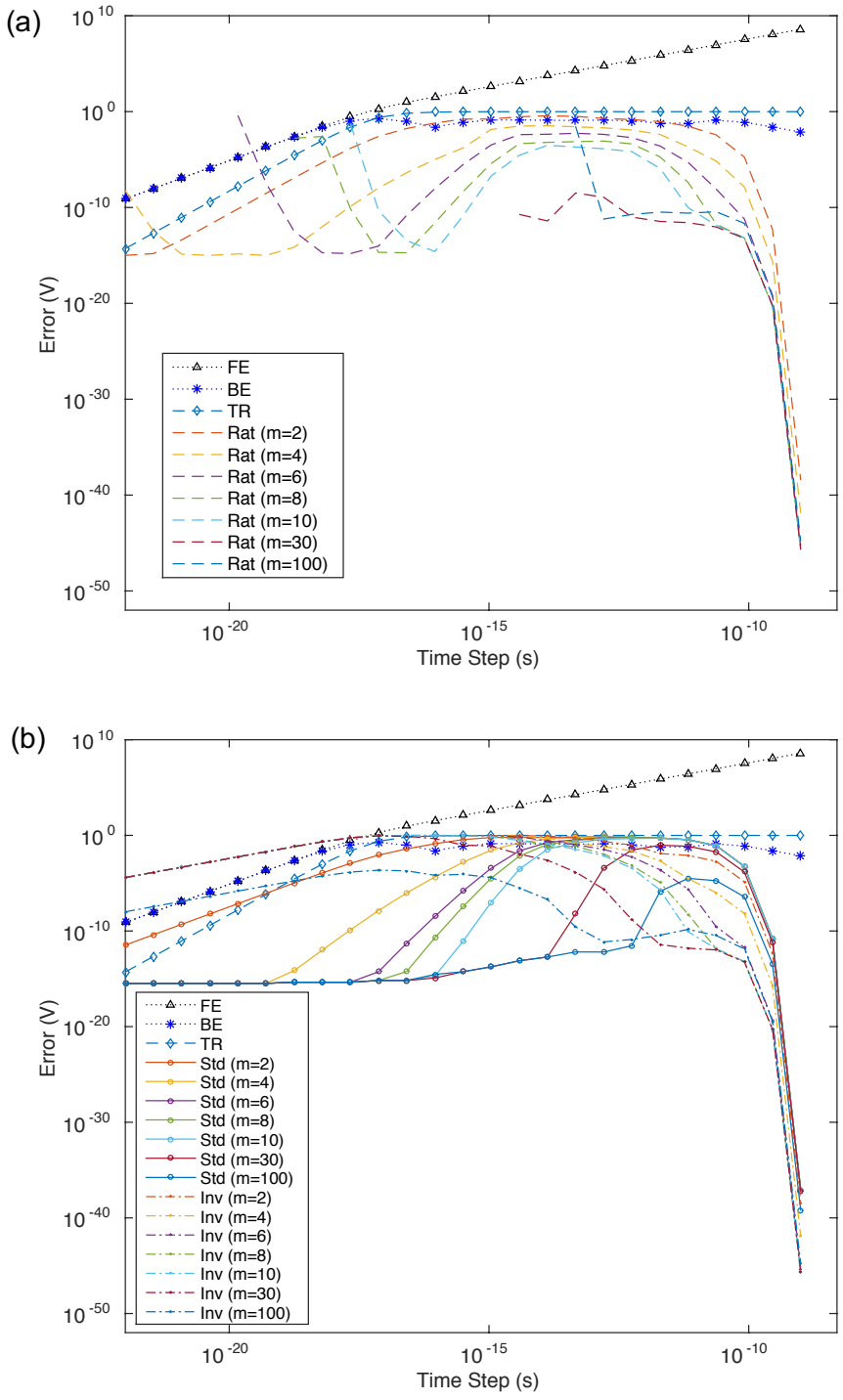

Figure 2.6: RC circuit's error distribution of the one-step integration results via different linear integrators with the same initial vector $x(0)$ and different time step h. (a) Rat vs. FE, BE, and TR; (b) Std and Inv vs. FE, BE, and TR.

Chapter 3

Stability Analysis of Matrix Exponential Based Integration Methods

3.1 Motivation

In modern very large-scale integration (VLSI) designs, the performance of power deliver network (PDN) has become a critical issue. The power supply from the package down to on-chip integrated circuits is distributed through metal layers and vias, which could be modeled as a linear network consisting of resistors, capacitors and inductors [38]. The on-chip circuit modules are simplified as time-varying current sources in PDN analysis. Due to the shrinking feature size and increasing design complexity, the network could easily consist of millions to billions of elements which result in an extremely huge system. Moreover, the values of elements in a system level PDN may vary greatly and the transient responses include many different scaled time constants. In order to characterize the long term dynamic behavior, an extended time span at small scaled time steps is necessary

and extra computation efforts are required. At the same time, the stiffness of system is increased which degrades the performance of traditional simulation methods. All the challenges make a fast and accurate simulator in high demand.

The widely accepted backward Euler (BE) and Trapezoidal (TR) usually serve as the baseline in traditional linear multi-step integration methods since they are proved to be *A-stable* [8, 34, 37]. However, solving a linear system is required at each time step and the performance of the implicit integration methods are impacted by the local truncation error (LTE).

A matrix exponential based integration method for PDN transient simulation is considered [6, 66]. Compared to the traditional linear multi-step methods, the matrix exponential based method is not bounded by the Dahlquist stability barrier thus the step size breaks the limitation of LTE [57, 66]. It has been explored with the efficient evaluation of matrix exponential and vector product (MEVP) via Krylov subspace method, which is considered as a high order polynomial approximation [33, 47]. The stability of matrix exponential based method when applied to ODEs has been well established in previous work [58, 66]. For general circuit simulation with DAEs, the stability remains an interesting topic [15, 27, 52, 60]. Numerical stability issues are reported in [6, 55] and reveal the limitation of MEVP computations with Krylov subspace. Similar problem occurs in the eigenvalue problems [30, 41] and model order reduction for interconnect simulation [45, 50] where Krylov subspace methods are widely used.

The aforementioned studies evoke the exploration of stability ensured algorithms. In this section, we focus on the stability of solving DAEs for PDN transient simulations. We adopt the rational Krylov subspace to efficiently capture the dominant dynamical behaviors of PDNs due to its flexibility to confine the spectrum of ill-conditioned systems [3, 13, 41, 53]. In Section 3.2, numerical stability problems will be reported on a general linear circuit which is originated from the singularity of system. The formulation of PDN transient simulation

provides a special format of DAEs, named as *semi-explicit DAEs with index-one* [27]. Inspired by the special structure of DAEs, we devise the implicit regularization to handle the singular matrices.

In Section 3.3, we prompt a modified Arnoldi algorithm with structured orthogonalization to construct the Krylov subspace for the calculation of matrix exponential. The orthonormal basis of Krylov subspace is generated by a quadratic norm with the capacitance matrix. The condition of original system is preserved which provides stable computations of MEVPs. Inspired by the structure of semi-explicit DAEs, the implicit regularization, as demonstrated in Section 3.2.3, is incorporated to handle the singular systems. Considering some extreme cases, we find that the spurious eigenvalues might exist in the results of Arnoldi and cause non-negligible error. A numerical pruning step is applied to eliminate the potential spurious eigenvalues after Arnoldi. The algorithm is validated with theoretical proof and experimental results.

3.2 Stability of Matrix Exponential Based Integration Methods for PDN Transient Simulation

3.2.1 Formulation of Semi-Explicit DAEs for PDNs

In Section 2, the formulation of circuit transient simulation is applicable for PDN transient simulation. The linear system is expressed as Eq. 2.3 which can be solved analytically with a nonsingular $\mathcal{C}$ matrix and PWL input. For a singular $\mathcal{C}$, extra regularization step will be needed for standard Krylov subspace method [7, 58]. In invert and rational Krylov methods, the construction of Krylov subspace does not require the computation of $\mathcal{C}^{-1}$. Instead, we calculate $\mathcal{G}^{-1}$ an $(\gamma\mathcal{G}+\mathcal{C})^{-1}$, respectively. For details see Section 2.4.1 and 2.4.2. We also notice that in Eq. 2.8, the $\mathcal{C}^{-1}$ is canceled by the $\mathcal{A}^{-1}$ in the initial

vectors of MEVPs

$$\widetilde{g}(t) = \mathcal{A}^{-1} g(t) = x(t) - \mathcal{G}^{-1} B u(t), \tag{3.1}$$

and

$$\frac{d\widetilde{\widetilde{b}}(t)}{dt} = \mathcal{A}^{-2} \frac{db(t)}{dt} = \mathcal{G}^{-1}\mathcal{C}\mathcal{G}^{-1} B \frac{du(t)}{dt}. \tag{3.2}$$

In most cases, the matrix $\mathcal{G}$ is invertible. We are able to avoid the extra regularization of DAEs. Although the different integration methods may not require the computation of $\mathcal{C}$ inverse, whether the formulation can produce the exact solution of DAEs remains to be explored. Actually, if we can obtain the generalized eigenvalues and corresponding eigenvectors for matrix pencil $(-\mathcal{G}, \mathcal{C})$, the solution can be derived based on [59]

Lemma 3.2.1. Considering a homogeneous system

$$\mathcal{C}\frac{dx}{dt} = -\mathcal{G}x,$$

u and λ are the eigenvector and eigenvalues of matrix pencil $(-\mathcal{G}, \mathcal{C})$, then

$$x = e^{t\lambda} u$$

is a solution of the system.

According to the relation in Section 2.3, the Hessenberg matrix contains the dominant eigenvalues of the system. Our target is to figure out whether the eigenvalues of original system are accurately captured by the Arnoldi algorithm, and whether there exist any spurious eigenvalues in the Hessenberg matrix that may cause numerical issues.

For general cases, a regular $\mathcal{C}$ matrix cannot always be achieved. The nodes without

nodal capacitance or inductance contribute to the algebraic equations of DAEs. The resulting singular $\mathcal{C}$ matrix makes direct computation of its inverse infeasible. However, we notice that the DAEs of a PDN follow a special structure and the analytical solution is still accessible. Following the *semi-explicit* DAEs with *differentiation index one* [27, 60], we are able to extract the underlying ODEs from the original Eq. 2.3.

Definition 3.2.1. Semi-explicit DAEs with Index-one. The structure requires

1. an invertible partial capacitance matrix $\widetilde{\mathcal{C}}$ including all the nonzero capacitance and inductance

2. invertible $\mathcal{G}$ as well as its partial matrix $\mathcal{G}_{11}, \mathcal{G}_{22}$

Then we have the semi-explicit DAEs

$$\begin{pmatrix} \widetilde{\mathcal{C}} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} + \begin{pmatrix} \mathcal{G}_{11} & \mathcal{G}_{12} \\ \mathcal{G}_{21} & \mathcal{G}_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} U_1 \\ U_2 \end{pmatrix}. \tag{3.3}$$

The state vector x in Eq. 3.3 is separated into the differential part x_1 for ODEs and algebraic part x_2. The input sources $U(t) = Bu(t)$ are reformatted in the same way. In general PDNs, the matrix $\mathcal{G}$ is positive semi-definite and invertible, and $\mathcal{C}$ is symmetric and diagonal. Let $\mathcal{C} = V_R \widetilde{\mathcal{C}} V_R^\top$ be the eigenvalue decomposition of $\mathcal{C}$ excluding the singular values. The whole space is introduced as the composition of the real basis and nullspace $V := \begin{bmatrix} V_R, V_N \end{bmatrix}$, so that the state variables can be decomposed as

$$x(t) = V_R x_1(t) + V_N x_2(t). \tag{3.4}$$

The semi-explicit DAEs in Eq. 3.3 can be transformed to explicit ODEs

$$\widetilde{\mathcal{C}}\dot{x}_1 + (\mathcal{G}_{11} - \mathcal{G}_{12}\mathcal{G}_{22}^{-1}\mathcal{G}_{21})x_1 = U_1 - \mathcal{G}_{12}\mathcal{G}_{22}^{-1}U_2 \tag{3.5}$$

$$\mathcal{G}_{21}x_1 + \mathcal{G}_{22}x_2 = U_2. \tag{3.6}$$

The underlying ODEs can be solved analytically with guaranteed stability. The algebraic equations on the second row can be calculated without accuracy loss. However, the sparsity of original system is lost. The dense matrices make the explicit ODEs computationally expansive. Regarded as ill-conditioned systems, the DAEs require appropriate techniques to generate stable results while preserving the sparsity of original matrices.

3.2.2 Stability Problems and Sensitivity Analysis of Numerical Integration Methods

We start from a one tank lumped RLC model as shown in Fig. 3.1. A step input current source I_S with rise time TR= $1ps$ is applied. The DAEs of the one tank RLC follow the semi-explicit structure as expressed in Eq. 3.7. The node voltages and branch currents in the state vector are marked in Fig. 3.1.

$$\begin{pmatrix} 0 & & & \\ & 0 & & \\ & & C_1 & \\ & & & L_1 \end{pmatrix} \begin{pmatrix} \dot{v_1} \\ \dot{v_2} \\ \dot{v_3} \\ \dot{i_L} \end{pmatrix} + \begin{pmatrix} \frac{1}{R_1} + \frac{1}{R_2} & -\frac{1}{R_1} & & \\ -\frac{1}{R_1} & \frac{1}{R_1} & & 1 \\ & & 0 & -1 \\ & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ i_L \end{pmatrix} = \begin{pmatrix} I_{bias} \\ 0 \\ -I_S \\ 0 \end{pmatrix} \tag{3.7}$$

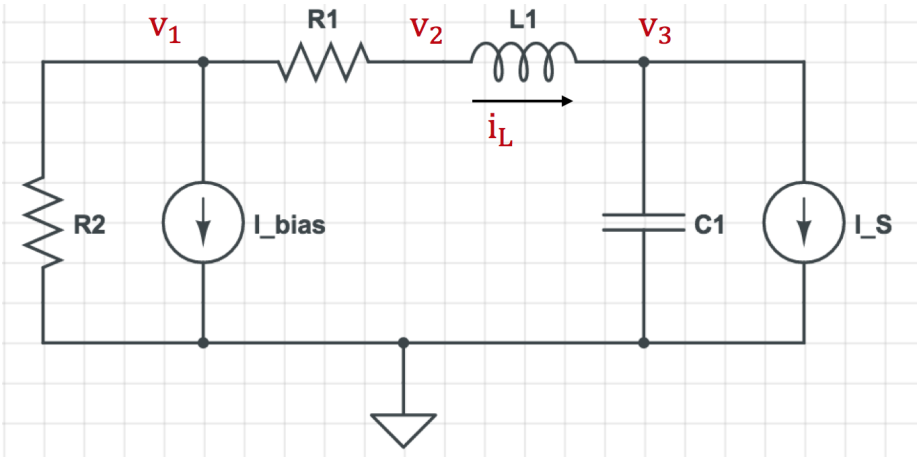

Figure 3.1: One tank RLC with $R1 = 100\mu\Omega, L1 = 0.5nH, C1 = 0.5nF$ and $R2 << R1$.

We recall the analytical solution Eq. 2.8 with MEVPs to solve the DAEs.

$$\begin{aligned} x(t+h) &= x(t) + term1 + term2, \\ term1 &= e^{h\mathcal{A}}\widetilde{g}(t) - \widetilde{g}(t), \\ term2 &= e^{h\mathcal{A}}\frac{\widetilde{\widetilde{db}}(t)}{dt} - \frac{\widetilde{\widetilde{db}}(t)}{dt} - h\frac{\widetilde{db}(t)}{dt}, \end{aligned} \tag{3.8}$$

where the input vectors of MEVPs are defined as Eq. 3.1 and Eq. 3.2. The vector $\frac{\widetilde{db}(t)}{dt} = \mathcal{A}^{-1}\frac{db(t)}{dt} = -\mathcal{G}^{-1}B\frac{du(t)}{dt}$. We only consider the implementation with matrix exponential function $e^{h\mathcal{A}}$, which is known as exponential related φ_0 function [24, 40]. The evaluation of MEVPs with other φ functions will be discussed in Chapter 4.

Rational Krylov subspace is constructed through Arnoldi process in the simulation to compute the MEVPs, as shown in Algorithm 2. We set the first step $h = 1ps$ for the input transition stage and use fixed step size for the stable stage. The accuracy of result $x(t+h)$ is reflected by the residual

$$residual(t+h) = \mathcal{C}\dot{x}(t+h) + \mathcal{G}x(t+h) - Bu(t+h), \tag{3.9}$$

Algorithm 2: Arnoldi algorithm for rational Krylov subspace

Input: $\mathcal{C}, \mathcal{G}, \gamma, mv$
Output: V_m, H_m

$v_1 = v/\|v\|$;
for $j = 1 : m$ **do**
  Solve $(\gamma\mathcal{G} + \mathcal{C})w = \mathcal{C}v_j$;
  for $i = 1 : j$ **do**
    $h_{i,j} = w^\top v_i$;
    $w = w - h_{i,j}v_i$;
  end
  $h_{j+1,j} = \|w\|$;
  $v_{j+1} = \frac{w}{h_{j+1,j}}$;
  if $r(h, m)$ ***PASS*** *residue check* **then**
    $m = j$;
    break;
  end
end

where the solution $x(t + h)$ and its derivative $\frac{dx(t+h)}{dt} = \frac{d}{dt}term1 + \frac{d}{dt}term2$ are based on Eq. 3.8 and the following approximation

$$\begin{aligned} term1 &\approx \beta_1 V_{m1} e^{h\frac{I-H_{m1}^{-1}}{\gamma}} e_1 - \widetilde{g}(t), \\ term2 &\approx \beta_2 V_{m2} e^{h\frac{I-H_{m2}^{-1}}{\gamma}} e_1 - \frac{d\widetilde{\widetilde{b}}(t)}{dt} - h\frac{d\widetilde{b}(t)}{dt}, \end{aligned}$$

and

$$\begin{aligned} \frac{d}{dt}term1 &\approx \beta_1 V_{m1} \frac{I - H_{m1}^{-1}}{\gamma} e^{h\frac{I-H_{m1}^{-1}}{\gamma}} e_1, \\ \frac{d}{dt}term2 &\approx \beta_2 V_{m2} \frac{I - H_{m2}^{-1}}{\gamma} e^{h\frac{I-H_{m2}^{-1}}{\gamma}} e_1 - \frac{d\widetilde{b}(t)}{dt}, \end{aligned}$$

where $\beta_1 = \|\widetilde{g}(t)\|$ and $\beta_2 = \|\frac{d\widetilde{\widetilde{b}}(t)}{dt}\|$. The variables $m1$ and $m2$ represent the dimension of converged Krylov subspace for term1 and term2, respectively. In the Arnoldi process, The residual $r(h, m)$ used to determine the convergence of MEVPs follows Eq. 2.32.

Fig. 3.2 depicts the node voltages and the solution residual in the simulation, showing that the residual terms on algebraic variables v_1 and v_2 start to increase at early stage and generally drive the whole system to an incorrect converging direction. Exact solution is calculated by PWL response from MATLAB continuous-time transfer function using fixed step size ($100ps$ for stable input). Trapezoidal (TRAP)[1] method results with fixed step size $100ps$ are plotted as comparison which show a deviation from exact solution as well.

The local error of one-step integration is accumulated and propagated to later simulation time. We start from the sensitivity analysis to explore how the origin and propagation of error. The sensitivity of the multi-step integration methods is defined as follows.

Remark 3.2.1. Use ϵ to denote the local error with input vector v, the perturbation of ϵ to the object function is approximated with a sensitivity matrix

$$F(v+\epsilon) - F(v) \approx D\epsilon \tag{3.10}$$

where $\epsilon \in \mathbb{R}^n$, $D \in \mathbb{R}^{n\times n}$ is the Jacobian matrix of F on ϵ.

Sensitivity of Trapezoidal Method

Trapezoidal time integration scheme is an implicit second-order method. If a numerical error exists in $x(t)$, it is propagated to the integrated result $x(t+h)$

$$(\frac{\mathcal{C}}{h} + \frac{\mathcal{G}}{2})x(t+h) = (\frac{\mathcal{C}}{h} - \frac{\mathcal{G}}{2})(x(t)+\epsilon) + B\frac{u(t)+u(t+h)}{2}. \tag{3.11}$$

[1]To distinguish from the input rise time TR, we use TRAP in short of Trapezoidal method.

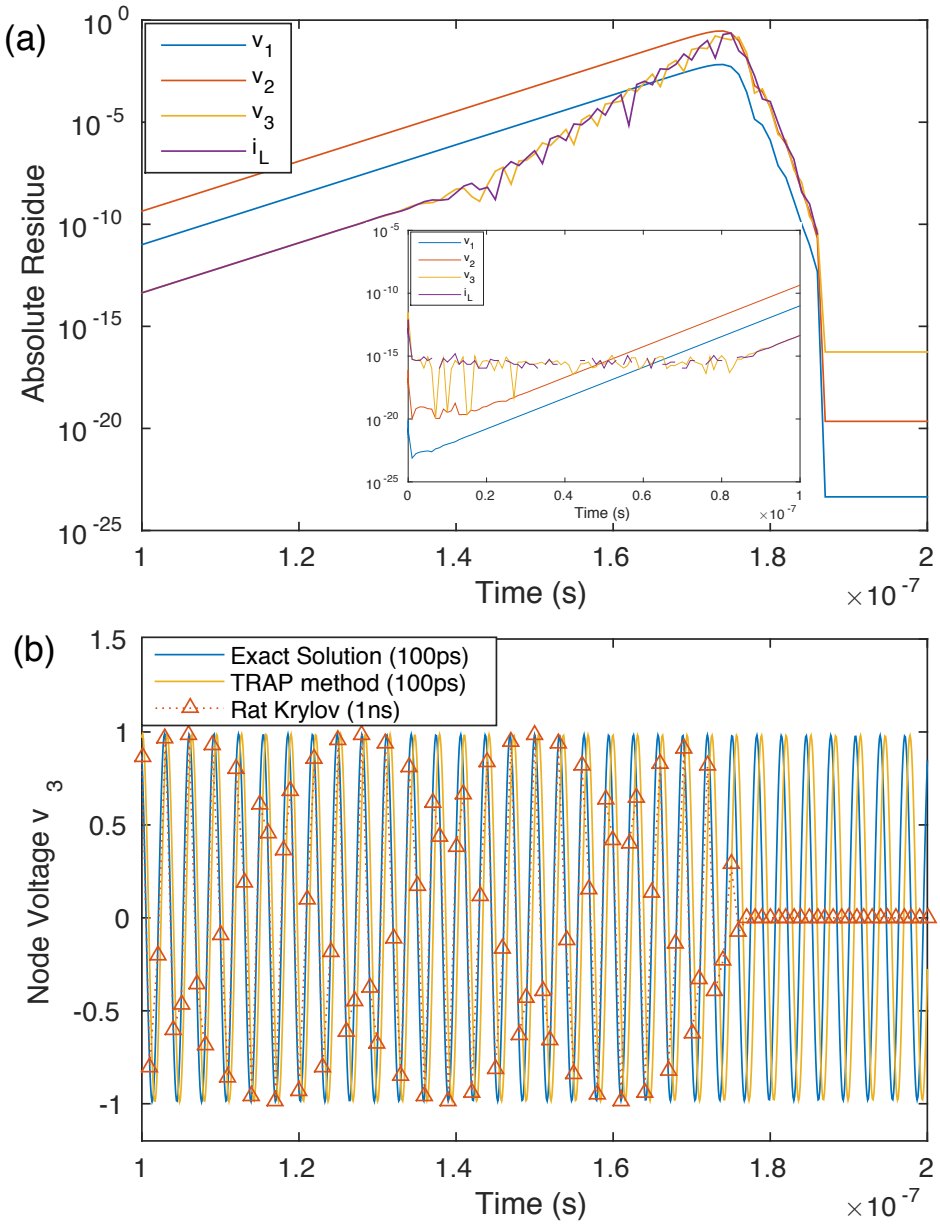

Figure 3.2: Simulation results of the one tank RLC. (a) absolute value of the residual for each variable in $x(t)$; (b) simulation results on v_3 with rational Krylov subspace method as well as TRAP method, exact solution is included as comparison.

Following the semi-explicit DAEs in Eq. 3.3, the sensitivity of solution at $t+h$ is derived as

$$\begin{aligned} Dx(t+h) &= (\frac{\mathcal{C}}{h}+\frac{\mathcal{G}}{2})^{-1}(\frac{\mathcal{C}}{h}-\frac{\mathcal{G}}{2}) = -\begin{pmatrix} \mathcal{G}_{11}+\frac{2\widetilde{\mathcal{C}}}{h} & \mathcal{G}_{12} \\ \mathcal{G}_{21} & \mathcal{G}_{22} \end{pmatrix}^{-1} \begin{pmatrix} \mathcal{G}_{11}-\frac{2\widetilde{\mathcal{C}}}{h} & \mathcal{G}_{12} \\ \mathcal{G}_{21} & \mathcal{G}_{22} \end{pmatrix} \\ &= \begin{pmatrix} -I+\frac{4}{h}\mathcal{G}_{+11}^{-1}\widetilde{\mathcal{C}} & 0 \\ \frac{4}{h}\mathcal{G}_{+22}^{-1}\mathcal{G}_{21}(\mathcal{G}_{11}+\frac{2\mathcal{C}}{h})^{-1}\widetilde{\mathcal{C}} & -I \end{pmatrix}, \end{aligned} \tag{3.12}$$

where the $\mathcal{G}_+$ denotes the matrix $\mathcal{G}+\frac{2\mathcal{C}}{h}$ and uses the subscript for its partial matrix. The upper part of D corresponds to the effects of error on the ordinary differential elements, while the lower part of D corresponds to the effects on the algebraic elements. We could tell that if an error exists in previous step, it might increase in the integrated results on x_1 and influence the error of x_2.

Sensitivity of Rational Krylov Method

Consider the stage when $u(t)$ is stable after the rise time, we only check MEVP term1 in the solution because the term2 is nonzero only in the first time step. The residual increases exponentially versus integration steps. The target function is approximated by

$$F(v_1) = e^{h\mathcal{A}}v_1 \approx V_m e^{h\frac{I-H_m^{-1}}{\gamma}}e_1, \tag{3.13}$$

where v_1 is the normalized input vector so that $\beta = 1$. Based on the orthogonalization process in Arnoldi algorithm, we have the relation between two neighboring basis vectors

$$h_{j+1,j}v_{j+1} = Mv_j - \Sigma_{i=1}^{j}h_{i,j}v_i, \tag{3.14}$$

where $M(\gamma) = (\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$ is used to construct the rational Krylov subspace. Assume the local error ϵ exists in v_1, it will be propagated to the whole subspace based on the relation

(3.14). For the one tank RLC circuit, the perturbation mainly contributes to the subspace basis V_m. We assume that the change of elements in H_m is negligible compared to that in V_m after introducing the error ϵ. The dimension m is at most 2 because it cannot exceed the rank of system, so we have

$$V_m = \begin{bmatrix} v_1 & v_2 \end{bmatrix}, H_m = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix}$$

and

$$h_{21}v_2 = \big(M(\gamma) - h_{11}I\big)v_1. \tag{3.15}$$

The derivation provides the sensitivity of rational Krylov subspace to the input local error ϵ

$$D\epsilon = \begin{bmatrix} \epsilon & h_{21}^{-1}(M(\gamma) - h_{11}I)\epsilon \end{bmatrix} \exp(h\frac{I - H_m^{-1}}{\gamma})e_1. \tag{3.16}$$

Finally we denote the matrix exponential $E_{xp} = \exp(h\frac{I-H_m^{-1}}{\gamma})$, the sensitivity matrix is given as

$$D = E_{xp,11}I + E_{xp,21}h_{21}^{-1}(M(\gamma) - h_{11}I). \tag{3.17}$$

If the magnitude of D diagonal terms is larger than 1, the perturbation of ϵ cannot be suppressed in later integration steps. Fig. 3.3 shows a well fitting of the sensitivity analysis and experimental phenomenon. For the one tank RLC circuit, the matrix D is diagonal dominant with the diagonal elements $D(1,1) = D(2,2)$. The slope in log-scale (over integration steps) of increasing residual is measured from simulation results under multiple choices of step sizes. We choose $\gamma = h/2$ so the slope is plotted as a function of γ.

To illustrate the effects of simulation parameters on the sensitivity analysis, Fig. 3.4

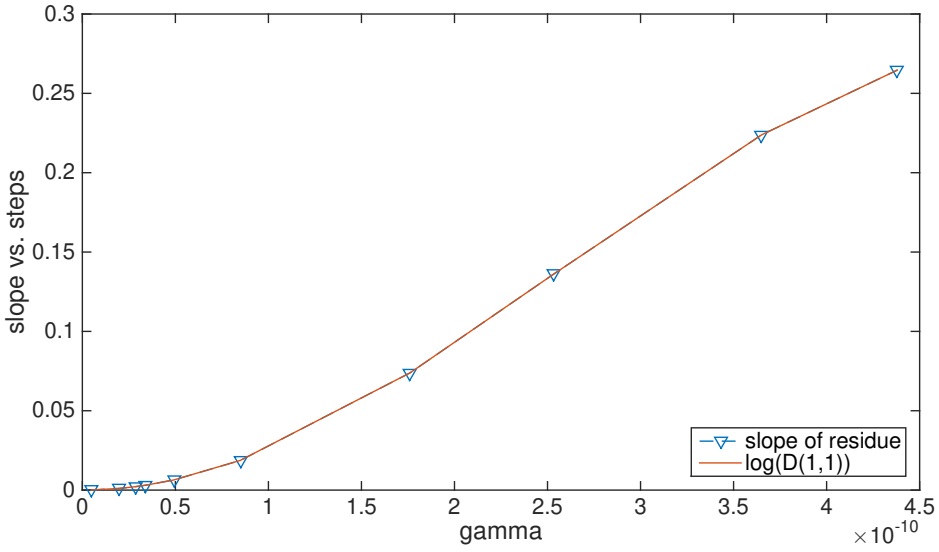

Figure 3.3: Slope of increasing residual versus $\gamma = \frac{h}{2}$ is well fitting its corresponding sensitivity D.

shows the distribution of $D(1,1)$ where the parameters h and γ are set as independent variables. The region where $|D(1,1)| > 1$ is plotted in red color. The first sensitivity term in Eq. (3.17) from the original local error in v_1 is determined by the matrix exponential versus h. The second term in Eq. 3.17 is affected by the exponential term and γ. Once a relatively smaller γ is used, M is close to an identity matrix so the coefficient $\frac{h_{11}}{h_{21}}$ becomes extremely large. For larger PDNs, the sensitivity analysis becomes more complicated and the Arnoldi process should be operated carefully to avoid the stability issues.

3.2.3 Implicit Regularization Approach

From the observations on ill-conditioned system from DAEs, the numerical error occurs in the calculation of algebraic variables and could result in stability issues in later simulation stage. For the linear systems of PDNs, $\mathcal{C}$ and $\mathcal{G}$ are positive semi-definite. The matrix $\mathcal{C}$ consisting of the ground capacitance and inductance is usually diagonal and symmetric. To eliminate the error in the nullspace $\mathcal{N}(\mathcal{G}^{-1}\mathcal{C}) = \mathcal{N}(\mathcal{C})$, the algebraic variables

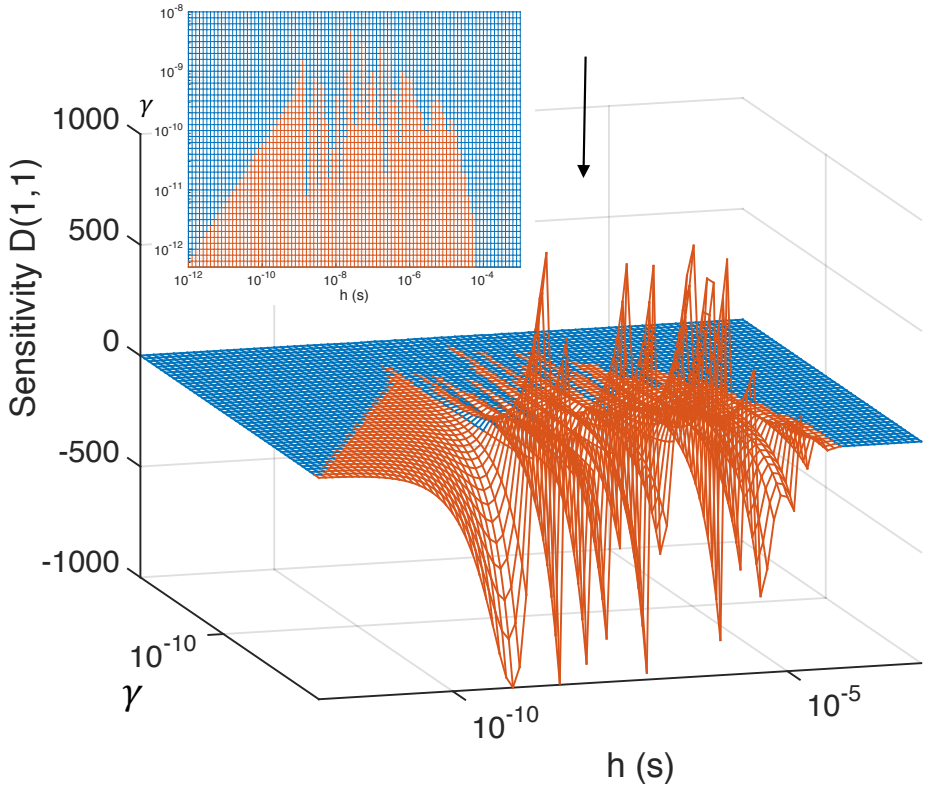

Figure 3.4: Sensitivity $D(1,1)$ of one tank RLC versus step size and γ. The red region shows $|D(1,1)| > 1$ and blue region $|D(1,1)| \leq 1$.

are set to zero in the Arnoldi process. The technique is called implicit regularization [6]

$$v = \begin{pmatrix} v_R \\ v_N \end{pmatrix} \Rightarrow v * I_R = \begin{pmatrix} v_R \\ v_N \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} v_R \\ 0 \end{pmatrix}. \tag{3.18}$$

where v_R corresponds to the variables in the underlying ODEs and v_N corresponds to the algebraic variables. The factor I_R only contains an identity matrix for the differential variables and zeros for the algebraic variables, similar to the separation of state vector in Eq. 3.3. The approach forces the computations in the range of $\mathcal{C}$, which is equivalent to solving the underlying ODEs implicitly.

Remark 3.2.2. The implicit regularization approach constitutes the solution of ODEs for

nonsingular variables x_1 in Eq. 3.3. Let $P = V_R V_R^\top$ be the orthogonal projection on the range of $\mathcal{C}$,

$$\begin{aligned} & P\mathcal{G}^{-1}\big(\mathcal{C}\dot{x} + \mathcal{G}x\big) = P\mathcal{G}^{-1}Bu \\ \Rightarrow & \begin{pmatrix} (\mathcal{G}^{-1})_{11}\widetilde{\mathcal{C}}\dot{x}_1 \\ 0 \end{pmatrix} + \begin{pmatrix} x_1 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathcal{G}_{11}^{-1}U_1 - \mathcal{G}_{11}^{-1}\mathcal{G}_{12}\mathcal{G}_{22}^{-1}U_2 \\ 0 \end{pmatrix}, \end{aligned} \tag{3.19}$$

where $(\mathcal{G}^{-1})_{11} = (\mathcal{G}_{11} - \mathcal{G}_{12}\mathcal{G}_{22}^{-1}\mathcal{G}_{21})^{-1}$ is partitioned matrix from $\mathcal{G}^{-1}$ and its inverse also serves as the conductance matrix of ODEs. The equation is equivalent to Eq. 3.5.

Proof. Applying $\mathcal{G}^{-1}$ on the original DAEs yields *one range consistency constraint on* $x(t)$ that the input vector of MEVP term1 $\widetilde{g}(t) = x(t) - \mathcal{G}^{-1}Bu(t)$ must lie in the range of $\mathcal{G}^{-1}\mathcal{C}$[2]. So we have the intermediate equation

$$\mathcal{G}^{-1}\mathcal{C}\dot{x} + x = \mathcal{G}^{-1}Bu.$$

By applying the implicit regularization, we project the vector in the range of $\mathcal{C}$ which is equivalent to multiplying P to the above equation and we get the another format of Eq. 3.5. Actually, the factor I_R in the implicit regularization is a special case of P when $\mathcal{C}$ is diagonal.

Next we need to prove that with the projected input vector, the rational Krylov subspace method still composes the approximation of exact solution of ODEs. In the

[2]Notice that the other input vector in term2 is also in the range of $\mathcal{G}^{-1}\mathcal{C}$.

construction of rational Krylov subspace, we multiply $(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$ to input vector and get

$$
\begin{aligned}
(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}v &= \begin{pmatrix} \gamma\mathcal{G}_{11}+\widetilde{\mathcal{C}} & \gamma\mathcal{G}_{12} \\ \gamma\mathcal{G}_{21} & \gamma\mathcal{G}_{22} \end{pmatrix}^{-1} \begin{pmatrix} \widetilde{\mathcal{C}} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} v_R \\ v_N \end{pmatrix} \\
&= \begin{pmatrix} (\gamma\widetilde{\mathcal{G}}+\widetilde{\mathcal{C}})^{-1} & * \\ * & * \end{pmatrix} \begin{pmatrix} \widetilde{\mathcal{C}}v_R \\ 0 \end{pmatrix} = \begin{pmatrix} (\gamma\widetilde{\mathcal{G}}+\widetilde{\mathcal{C}})^{-1}\widetilde{\mathcal{C}}v_R \\ * \end{pmatrix}. \qquad (3.20)
\end{aligned}
$$

The partial conductance matrix $\widetilde{\mathcal{G}} = \mathcal{G}_{11} - \mathcal{G}_{12}\mathcal{G}_{22}^{-1}\mathcal{G}_{21}$ and differential variables in v_R compose the exactly same ODEs to Eq. (3.5). By setting the algebraic variables to zeros in $v_N \in \mathcal{N}(\mathcal{C})$ in the Arnoldi process, the computation with original system is still valid. □

The updated Arnoldi with implicit regularization is shown in Algorithm 3. The extra matrix and vector multiplications don't increase the computation complexity of original algorithm. The algebraic variables in $x_2(t)$ could be solved algebraically. All the calculations are based on the original system and preserve the sparsity. Extraction of the underlying ODEs is unnecessary for solving DAEs of PDN transient simulation, and we avoid the computation of dense matrices after the explicit regularization.

Lemma 3.2.2. The regularized Arnoldi process excludes the calculation of x_2 and maintains the sparsity of original system.

Simulation results of the one tank RLC with implicit regularization are shown in Fig. 3.5, which fit the exact solution. Residuals of v_3 and i_L remain at a low level ($\approx 10^{-15}$) when the input current is stable. The other variable are solved algebraically and the system no longer suffers from the stability problem.

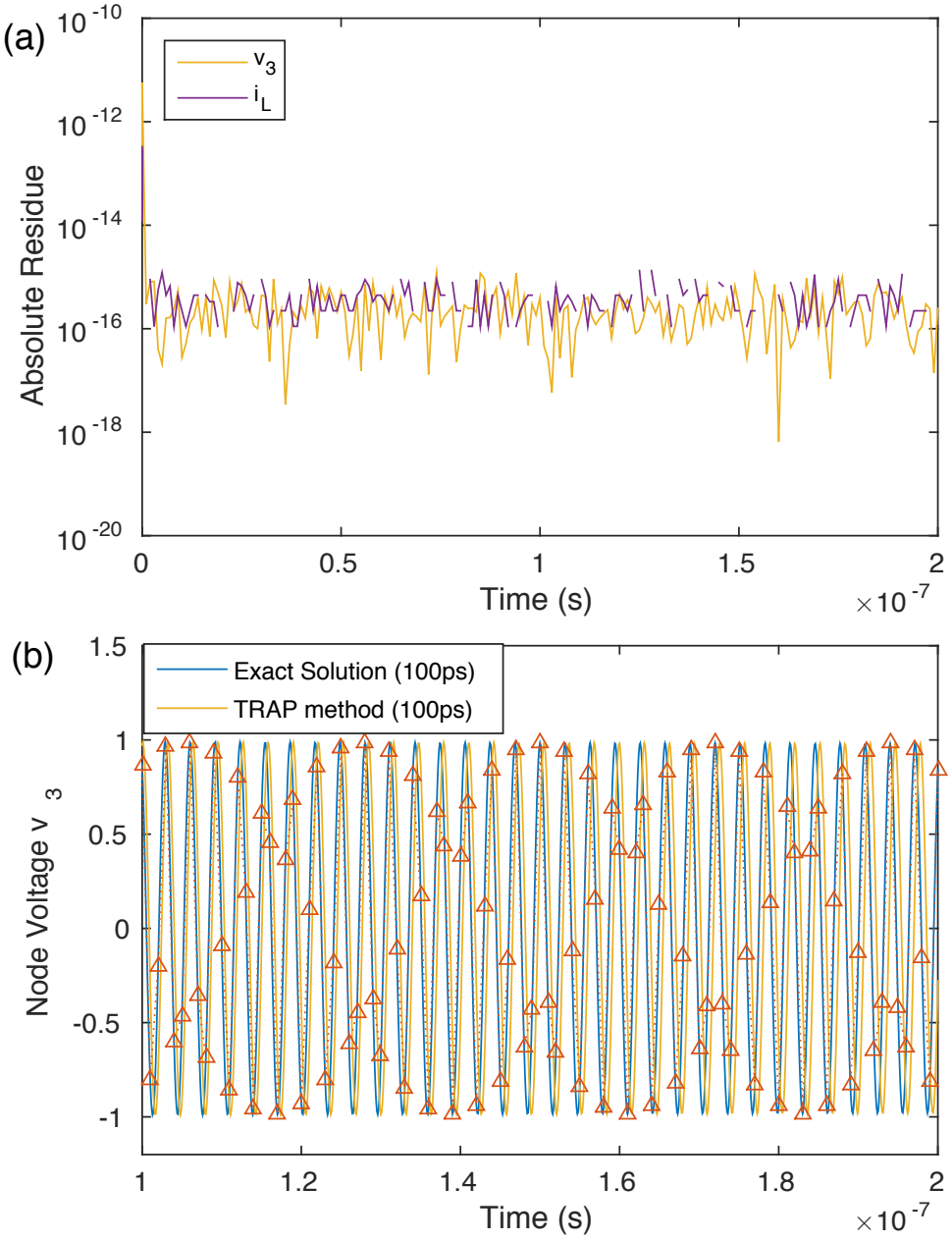

Figure 3.5: Simulation results of the one tank RLC with implicit regularization. (a) The absolute residual no longer increase and (b) simulation results well fit the exact solution. Node voltages v_1, v_2 are solved algebraically.

Algorithm 3: Arnoldi algorithm for rational Krylov subspace with implicit regularization

Input: $\mathcal{C}, \mathcal{G}, \gamma, m, v$
Output: V_m, H_m
$v_1 = v * I_R / \|v * I_R\|$;
for $j = 1 : m$ **do**
  Solve $(\gamma\mathcal{G} + \mathcal{C})w = \mathcal{C}v_j$;
  Set $w = w * I_R$;
  for $i = 1 : j$ **do**
    $h_{i,j} = w^\top v_i$;
    $w = w - h_{i,j}v_i$;
  end
  $h_{j+1,j} = \|w\|$;
  $v_{j+1} = \frac{w}{h_{j+1,j}}$;
  if $r(h, m)$ ***PASS*** *residue check* **then**
    $m = j$;
    break;
  end
end

3.3 A Stability Preserved Arnoldi Algorithm with Structured Orthogonalization

In this section, we introduce a new Arnoldi scheme with structured orthogonalization to generate one stable Krylov subspace and to compute the MEVPs. The orthogonality is based on the positive semi-definite matrix $\mathcal{C}$. The $\mathcal{C}$ semi-inner product plays a fundamental role in enforcing the numerical range of the operator in the left half plane, which satisfies the stability condition.

3.3.1 An Arnoldi Process with Structured Orthogonalization

The rational Krylov subspace were originally developed for computing eigenvalues and eigenvectors of large matrices [46]. The method is a very promising manner in computing the MEVP $e^{hA}v$, which can converge geometrically when the range of A is in

the left half complex plane [9]. Typically, the Krylov subspace approximation does not completely lie in the left half plane.

Recall the computation of MEVPs with rational Krylov subspace, as shown in Section. 2.4.2. By multiplying the transpose of orthonormal basis $V_m^\top$ to the left of the relation (2.30), we have

$$V_m^\top(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}V_m = H_m. \tag{3.21}$$

The spectrum of system is confined with γ and then inverted to produce eigenvalues with nonnegative real parts. The slow decaying and oscillation components become dominant in the rational Krylov subspace and can be quickly approximated. In the computation of MEVPs, the approximated eigenvalues are recovered by the operation on the Hessenberg matrix $\frac{I-H_m^{-1}}{\gamma}$. PDNs contain positive semi-definite $\mathcal{C}$ and $\mathcal{G}$ matrices, then a positive semi-definite system $(\gamma\mathcal{G}+\mathcal{C})^{-1}$ is generated according to Lemma 3.3.1 [50].

Lemma 3.3.1. If a real matrix A is positive semi-definite and B is symmetric, then BAB is positive semi-definite. Furthermore, if A is nonsingular then its inverse A^{-1} is also positive semi-definite. Apply to negative semi-definite matrices as well.

However, the whole system $(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$ doesn't necessarily be positive semi-definite. In order to generate stable results, firstly a positive semi-definite H_m should be ensured for the rational Krylov subspace. Otherwise, after the recovery operation potential positive eigenvalues may exist in the matrix exponential and cause stability issues. A new Arnoldi process is proposed in Algorithm 4, where we use $\mathcal{C}$ for the inner products to normalize the vectors in V_m. The implicit regularization is operated for singular matrices for semi-explicit DAEs.

Algorithm 4: Arnoldi algorithm with structured orthogonalization and implicit regularization

Input: $\mathcal{C}, \mathcal{G}, \gamma, v, h, m$
Output: H_m, V_m

Set $v = v * I_R$;
$v_1 = \frac{v}{\|v\|_{\mathcal{C}}}$ where $\|v\|_{\mathcal{C}} = \sqrt{v^\top \mathcal{C} v}$ and $v_1^\top \mathcal{C} v_1 = 1$;
for $j = 1 : m$ **do**
    Solve $(\gamma\mathcal{G} + \mathcal{C})w = \mathcal{C}v_j$;
    Set $w = w * I_R$;
    for $i = 1 : j$ **do**
        $h_{i,j} = w^\top * \mathcal{C} * v_i$;
        $w = w - h_{i,j} v_i$;
    end
    $h_{j+1,j} = \|w\|_{\mathcal{C}}$;
    $v_{j+1} = \frac{w}{h_{j+1,j}}$;
    if $r(h, m)$ ***PASS*** *residue check* **then**
        $m = j$;
        break;
    end
end

The semi-inner product with $\mathcal{C}$ orthogonality is defined as

$$\|v\|_{\mathcal{C}} = \sqrt{v^\top \mathcal{C} v}, \tag{3.22}$$

which provides the relation following the same expression to Eq. 2.30

$$(\gamma\mathcal{G} + \mathcal{C})^{-1}\mathcal{C}V_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^\top, \tag{3.23}$$

with $V_m^\top \mathcal{C} V_m = I$. To illustrate the property of the corresponding subspace, we multiply $V_m^\top \mathcal{C}$ to the left of Eq. 3.23

$$V_m^\top \mathcal{C}(\gamma\mathcal{G} + \mathcal{C})^{-1}\mathcal{C}V_m = H_m. \tag{3.24}$$

From Lemma 3.3.1, the generation of rational Krylov subspace is performed with a positive

semi-definite matrix $\mathcal{C}(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$. Theorem 3.3.1 indicates that the passivity property of system with $\mathcal{C},\mathcal{G}$ are preserved in the Hessenberg matrix H_m [42, 50].

Theorem 3.3.1. *Given A such that $A+A^\top$ is negative semi-definite, Arnoldi algorithm is used to construct the Krylov subspace V_m and Hessenberg matrix H_m*

$$V_m^\top A V_m = H_m, \tag{3.25}$$

then passivity of original matrix is preserved.

In rational Krylov subspace, the spectrum of $\mathcal{A}$ is shifted and inverted to be positive with ratio γ. It could be proved that the H_m generated by Algorithm 4 preserves the positive semi-definiteness. The Algorithm 4 improves the performance of H_m on approximating the dominant eigenvalues and prevent the results from the contamination of ill-conditioned system. Although the construction of Krylov subspace is implicitly projected to the range of $(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$ where the nullspace vector is eliminated, the nonzeros in the input vector are introduced through the orthogonalization. So the implicit regularization step is still necessary. Compared to the original Arnoldi, the quadratic norm with $\mathcal{C}$ costs no extra computation because the matrix and vector multiplication is already calculated and can be reused, as described in line 4 of Algorithm 4. The implicit regularization takes an extra multiplication in each iteration and totally extra mN^2 computations. The computation cost of the new Arnoldi algorithm is similar to the original Arnoldi process, which is $O(mN^2)$.

3.3.2 Numerical Pruning of Spurious Eigenvalues

With structured orthogonalization, a guaranteed positive semi-definite H_m is created. However, in some corner cases the recovery of original spectrum with $\frac{I-H_m^{-1}}{\gamma}$ encounters numerical issues. It is observed that spurious eigenvalues might exist in H_m if $\mathcal{C}$ is ill-conditioned. It is necessary to figure out the situation before the error contaminates the

results.

We decompose H_m into the Jordan canonical form

$$H_m = U \begin{pmatrix} \Sigma & 0 \\ 0 & N \end{pmatrix} U^{-1}. \tag{3.26}$$

We define λ as the nonzero eigenvalues of the matrix pencil $(-\mathcal{G}, \mathcal{C})$. The diagonal matrix Σ includes the nonzero eigenvalues which tend to approximate $\frac{1}{1+\gamma\lambda}$. The spectrum of the Hessenberg matrix H_m does not necessarily lie in the shifted and inverted range of $\lambda\big((\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}\big)$. Assume θ is an eigenvalue of H_m and w is the corresponding eigenvector, we multiply w to the relation Eq. 3.23

$$\begin{aligned} (I-\gamma\mathcal{A})^{-1}V_m w &= V_m H_m w + h_{m+1,m} v_{m+1} e_m^\top w \\ &= \theta V_m w + h_{m+1,m} w(m) v_{m+1}. \end{aligned} \tag{3.27}$$

where $w(m)$ denotes the mth element of the vector. If the residual term $h_{m+1,m}w(m)v_{m+1}$ is small enough, then θ and $V_m w$ are the eigenvalue and corresponding eigenvector of matrix $(\gamma\mathcal{G}+\mathcal{C})^{-1}\mathcal{C}$. The exact eigenvalues of the system are interpolated by Arnoldi process. Once the residual is not negligible, we need to be careful using H_m.

The other part N, known as nil-potent, corresponds to the spurious eigenvalues that will contaminate the results. Besides, if the range of matrix pencil $(\mathcal{G}, \mathcal{C})$ is briefly approximated in advance, upper and lower bounds could be set to avoid huge deviation in the eigenvalues of H_m. The spurious eigenvalues in N are set to zeros, we call this process *pruning* of eigenvalues. After pruning, Drazin inverse is applied to evaluate H_m^{-1} [51].

Definition 3.3.1. Drazin Inverse of a Singular Matrix If we have a matrix that is decomposed into Jordan canonical form, as shown in Eq. 3.26. Then its Drazin inverse $H_m^\dagger$

is computed with

$$H_m^{\dagger} = U \begin{pmatrix} \Sigma^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^{-1}. \tag{3.28}$$

Therefore, the calculation of MEVPs takes Drazin inverse of H_m instead of exact matrix inverse. Previous work [51] proves the analytical solution with ODEs is applicable to ill-conditioned $\mathcal{C}$ if and only if the input vector lies in the range of system matrix. With the implicit regularization techniques, the space spanned by basis vectors is in the range of system matrix.

3.3.3 Numerical Experiments on RC and RLC Networks

To validate the new Arnoldi algorithm with structured orthogonalization, we adopt a general RC mesh as well as an RLC network and operate a one step integration with rational Krylov subspace for the transient simulation of both cases. The properties of the circuits and design of experiments are discussed in detail in this section. The test cases will be used in later explorations for potential performance improvements.

Case 1: performance on RC network

We start from a general 10×10 RC mesh consisting of 100 resistors and 100 capacitors. Each node in the RC network is connected by a self capacitor to ground. The matrices $\mathcal{C}, \mathcal{G}$ are positive definite, symmetric and nonsingular. The eigenvalues of $\mathcal{G}^{-1}\mathcal{C}$ lie in the range of $[10^{-18}, 10^{-11}]$.

Given a zero initial state $x(0)$ under all zeros input at time $t = 0$, we assume that the input source keeps increasing within the simulation time with unchanged slope. The analytical solution with MEVPs is valid for a one step integration to get the results $x(h)$ with step size h. The transient response of the RC mesh only MEVP term2 as derived

from Eq. 3.8

$$x(h) = e^{h\mathcal{A}}\widetilde{\widetilde{v}} - \widetilde{\widetilde{v}} - h\widetilde{v} \approx \beta V_m e^{h\frac{I-H_m^{-1}}{\gamma}} e_1 - \widetilde{\widetilde{v}} - h\widetilde{v}, \tag{3.29}$$

where the input vector $\widetilde{\widetilde{v}}$ is an abbreviated notation of vector $\frac{\widetilde{\widetilde{b}(t)}}{dt}$ in Eq. 3.2. The full expression is equivalent to the MEVP term2. Notice that the norm in structured orthogonaliazation is expressed as $\beta = ||\widetilde{\widetilde{v}}||_{\mathcal{C}}$.

The ratio γ is set as $h/2$ empirically. Benefiting from the symmetry of the original system matrices and structured orthogonalization, the Lanczos algorithm is used to save computations [53]. The MEVP in solution is evaluated at different step sizes with increasing dimension of Krylov subspace. Exact solution is computed with the explicit matrix $\mathcal{A}$ using MATLAB *expm* function as reference. Fig. 3.6 includes the absolute error distribution of results using Algorithm 4 (implemented as Lanczos) and the original Arnoldi.

$$err_{abs} = \left|\left|\beta V_m e^{h\frac{I-H_m^{-1}}{\gamma}} e_1 - e^{h\mathcal{A}} v\right|\right|. \tag{3.30}$$

In Fig. 3.6 (a) the region with spurious eigenvalues is plotted in red color where the error could grow extremely high with original Arnoldi algorithm, while the issue is resolved by the structured orthogonalization.

Case 2: performance on RLC network

We introduce a PDN circuit constructed by 260 resistors, 160 capacitors and 160 inductors, which leads to a singular $\mathcal{C}$ and nonsymmetric $\mathcal{G}$ matrices with size $n = 507$. The spectrum of $\mathcal{G}^{-1}\mathcal{C}$ lies in the range of $-[10^{-16}, 10^{-8}]$. The existence of inductors introduce complex eigenvalues with conjugate pairs to the system.

Same experiment is operated on the RLC network. The computation of MEVPs is concluded in Eq. 3.29. Implicit regularization technique is used to handle the singularity.

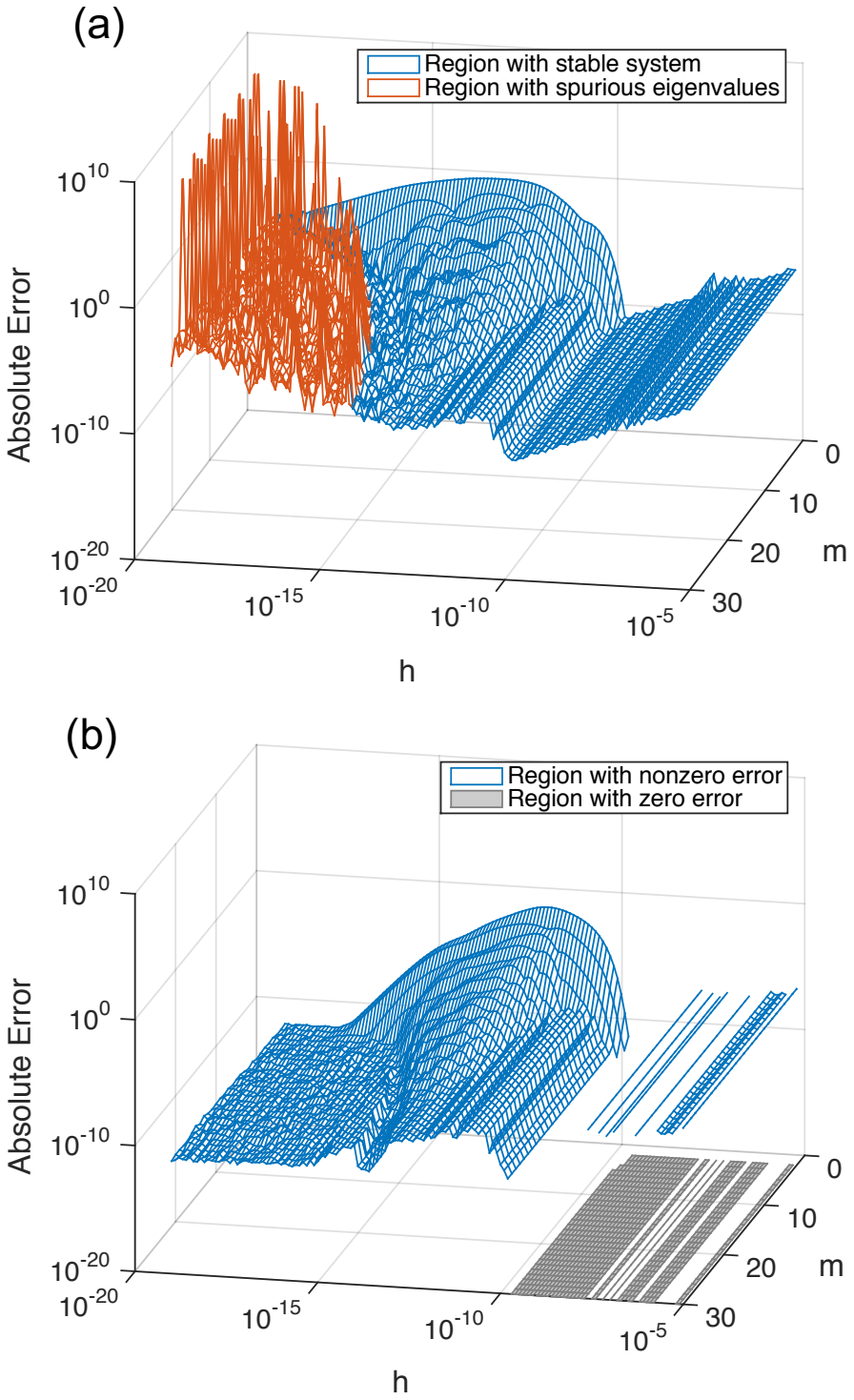

Figure 3.6: RC network with n=100: the absolute error err_{abs} with φ_0 function is computed versus h and m with (a) original Arnoldi; (b) Lanczos plus structured orthogonalization.

The exact solution is computed by explicitly solving the ODEs according to Eq. 3.5. The discontinuity in Fig. 3.7 is from the *NaN* result suffering from numerical issues. Compared to the original Arnoldi, structured orthogonalization improves the stability of results significantly, as shown in Fig. 3.6. When the step size is relatively small, the results are dominated by the numerical precision errors. In case of any spurious eigenvalues in the simulation of RLC network, a numerical pruning process is implemented to clean the unwanted spurious eigenvalues as explained in Section 3.3.2.

More error analysis on the computation of MEVPs can be found in Sec. 4.1.1.

3.4 Summary

In this chapter, we apply the matrix exponential based integration method to the PDN transient simulation. The MEVPs are evaluated with rational Krylov subspace and the numerical stability issues are observed with the ill-conditioned systems of PDNs. Considering the structure of semi-explicit DAEs with index one, an implicit regularized Arnoldi algorithm is devise to handle the singular matrices and solve the underlying ODE system anylytically. The sparsity of original matrices is preserved. We propose a stability guaranteed Arnoldi algorithm with structured orthogonalization, where the orthogonality is induced from the singular system matrix $\mathcal{C}$. The We validate the performance of the new algorithm on typical power grids (RC and RLC networks) transient simulation and observe significant improvements with error analysis.

Chapter 3 is a combination of the material in the following two works: P. Chen, C. K. Cheng, D. Park, and X. Wang, "Transient circuit simulation for differential algebraic systems using matrix exponential," in *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018. X. Wang, P. Chen, and C.-K. Cheng, "Stability and convergency exploration of matrix exponential integration on power delivery

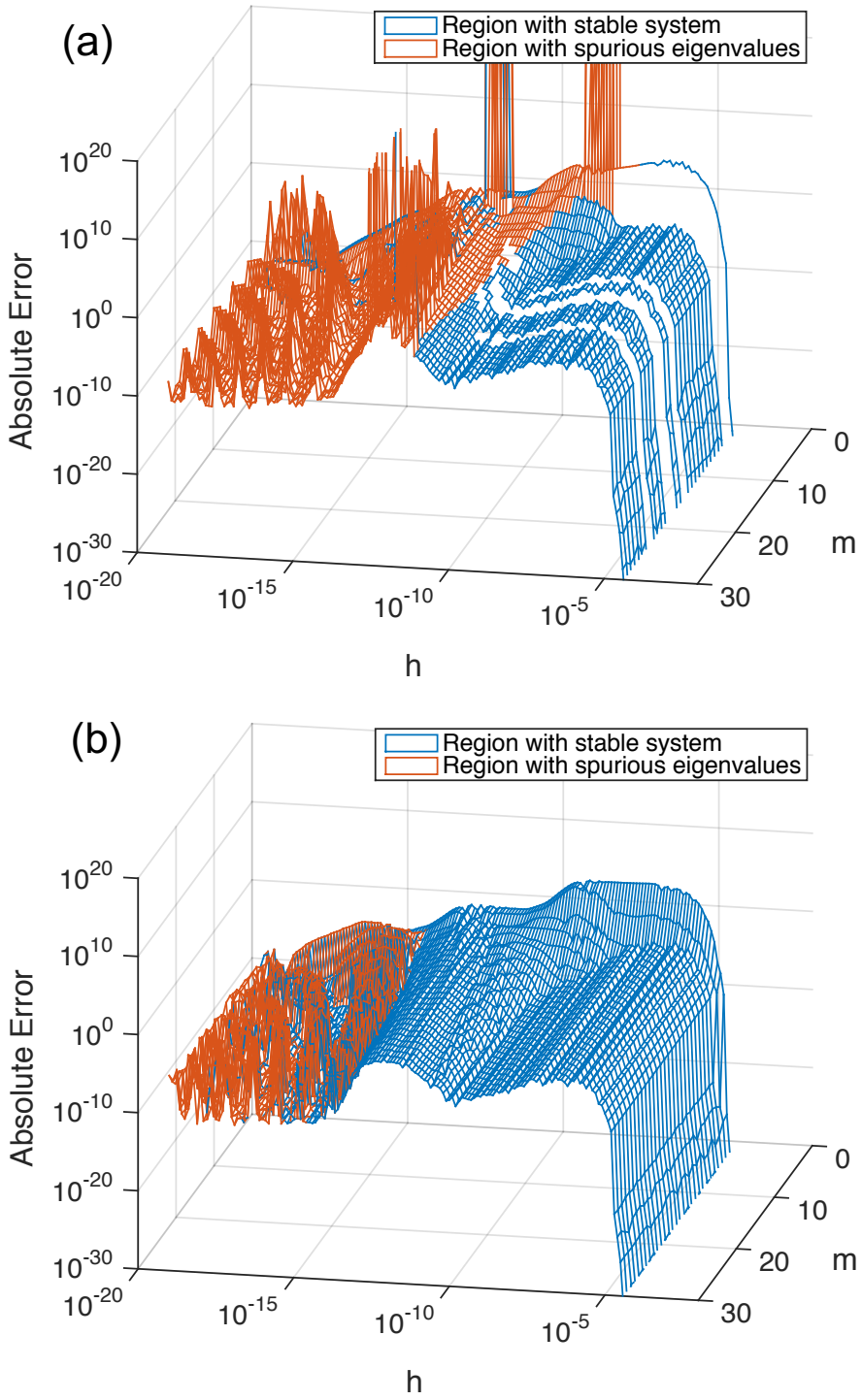

Figure 3.7: RLC network with size n=507: the absolute error err_{abs} versus h and m is plotted with (a) original Arnoldi (b) Arnoldi plus structured orthogonalization. The reference solution is from the explicit calculation of underlying ODEs.

network transient simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019. The author is the primary author and investigator of the papers. The chapter also contains the content the work: P. Chen, C.-K. Cheng, and X. Wang, "Arnoldi algorithms with structured orthogonalization," *arXiv preprint arXiv:2005.14468*, 2020. The author is one of the primary authors and investigators of this work.

Chapter 4

Numerical Performance of Exponential Integrators on System-Level PDN Simulations

Based on the stable rational Krylov subspace generated by the Arnoldi algorithm with structured orthogonalization, we are able to apply the new method to the transient simulation framework of PDNs. In this chapter, we further improve the accuracy and convergence rate of the MEVPs by exploring the exponential related φ functions. Besides, the ratio γ which is used to confine the spectrum of original system is further studied. We integrate the advanced techniques into the total framework for general system-level PDNs and report the simulation results.

4.1 Numerical Performance of MEVPs with φ Functions

In previous sections, we try to solve the solution with MEVPs expressed with the matrix exponential $e^{h\mathcal{A}}$. However, we notice that the MEVPs in the Eq. 2.8 contain the format of $(e^{h\mathcal{A}} - I)\mathcal{A}^{-1}$ which may suffer from rounding errors [20]. Consider a simple function

$$f(z) = (e^z - 1)/z = \sum_{i=0}^{\infty} z^i/(i+1)!, \tag{4.1}$$

where z is a real number. The direct calculation of $f(z)$ suffers severe cancellation when $|z| \ll 1$. We implement the computation of $f(z)$ with different methods in MATLAB. As shown in the second column of Table. 4.1, the direct method cannot provide an accurate evaluation for $z \approx 0$ in floating point precision.

Table 4.1: Computed $f(z) = (e^z - 1)/z$ via Different Methods

z	Direct Method	Augmented φ Method
10^{-5}	1.000005000006965	1.000005000016667
10^{-6}	1.000000499962184	1.000000500000167
10^{-7}	1.000000499962184	1.000000050000002
10^{-8}	0.999999993922529	1.000000005000000
10^{-9}	1.000000082740371	1.000000000500000
10^{-10}	1.000000082740371	1.000000000050000
10^{-11}	1.000000082740371	1.000000000005000
10^{-12}	1.000088900582341	1.000000000000500
10^{-13}	0.999200722162641	1.000000000000050
10^{-14}	0.999200722162641	1.000000000000005
10^{-15}	1.110223024625156	1.000000000000000
10^{-16}	0	1.000000000000000

Here we introduce another method with the definition of exponential related φ

functions [24, 40, 49]

$$\varphi_s(z) = \int_0^1 e^{(1-\theta)z} \frac{\theta^{s-1}}{(s-1)!} d\theta, \quad s \geq 1. \tag{4.2}$$

where the different order of φ_s functions follow the recursive relation

$$\begin{aligned} \varphi_0(z) &= e^z \\ \varphi_s(z) &= \frac{\varphi_{s-1}(z) - \frac{1}{(s-1)!}}{z}, \quad s \geq 1. \end{aligned} \tag{4.3}$$

Previous work [49] provides an augmented matrix method to cheaply compute the MEVPs with φ_s functions.

Theorem 4.1.1. *An augmented matrix is used to compute the MEVPs with different orders of φ functions. Given a vector $c \in \mathbb{C}^m$ and a matrix $H_m \in \mathbb{C}^{m\times m}$, the augmented matrix is constructed*

$$\widetilde{H}_m = \begin{pmatrix} H_m & ce_1^T \\ 0 & E \end{pmatrix} \in \mathbb{C}^{(m+p)\times(m+p)}, \; E = \begin{pmatrix} e_2^T \\ \vdots \\ e_p^T \\ 0 \end{pmatrix} \in \mathbb{C}^{p\times p},$$

where $e_1, e_2, \ldots, e_p$ are unit vectors with $e_s(s) = 1$ for $s = 1, \ldots, p$. Then

$$exp(\tau\widetilde{H}_m) = \begin{pmatrix} \varphi_0(\tau H_m) & F \\ 0 & e^{\tau E} \end{pmatrix}. \tag{4.4}$$

For MEVP with any order of φ functions $1 \leq s \leq p$

$$F(1:n, s) = \tau^s \varphi_s(\tau H_m)c. \tag{4.5}$$

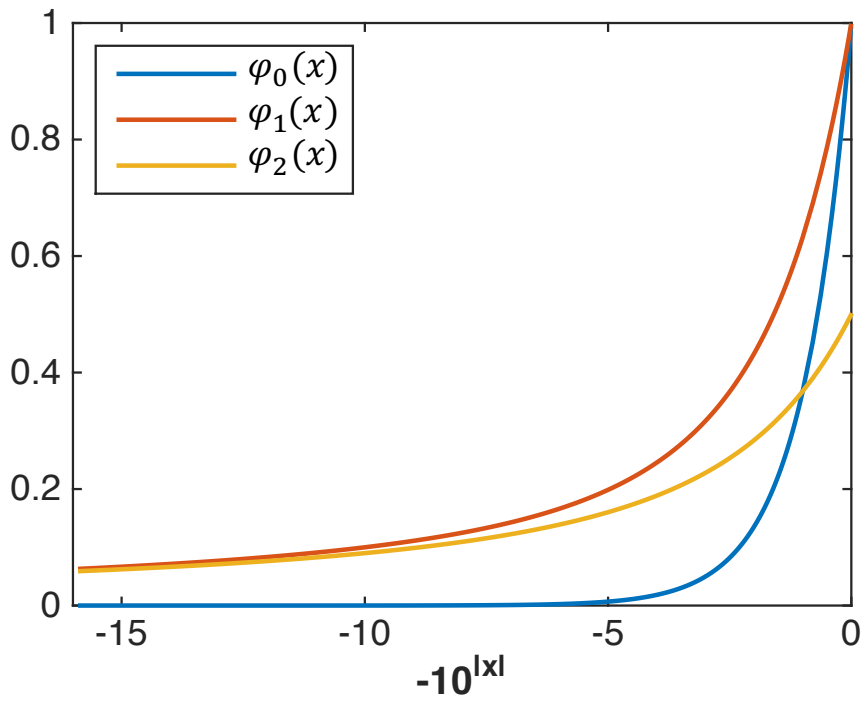

Figure 4.1: The curves of φ_0, φ_1, and φ_2 functions on the negative real axis. The magnitude of x is in log scale.

The function follows the definition of φ functions as $f(z) = \varphi_1(z)$, which could be calculated with the augmented matrix method as described in Theorem 4.1.1. We implement the matrix exponential with MATLAB *expm* function on the vector $c = [1]$ and set other parameters as $\tau = 1$. The results are shown in the third column of Table 4.1, where the numerical precision problem does not exist.

The performance of φ functions on the computation of MEVPs are closely related to the range of exponent. The passive systems of PDNs contain eigenvalues with negative real parts. In Fig. 4.1, the curves of multiple orders of φ functions are plotted on the negative real axis. The φ_s functions become smoother as s increases. The properties of φ_s functions will be used for error analysis when applied to the evaluation of MEVPs.

In order to avoid the potential numerical errors, we reformat the MEVPs in Eq. 2.8

with φ_s functions [24, 40, 56]

$$\Rightarrow x(t+h) = x(t) + h\varphi_1(h\mathcal{A})g(t) + h^2\varphi_2(h\mathcal{A})\frac{db(t)}{dt}, \tag{4.6}$$

where the input vectors are defined in Eq. 2.9 and 2.10. Following the relations in Eq. 4.3, we have alternative ways to evaluate the MEVPs. Notice that in Section 3.2 Eq. 3.8 we adopt the φ_0 function for both MEVP terms, we summarize the φ_s functions usage in the computations as

$$\begin{aligned}
\text{term1} &= h\varphi_1(h\mathcal{A})g(t) \\
&= \varphi_0(h\mathcal{A})\widetilde{g}(t) - \widetilde{g}(t), \qquad (4.7) \\
\text{term2} &= h^2\varphi_2(h\mathcal{A})\frac{db(t)}{dt} \\
&= h\varphi_1(h\mathcal{A})\frac{d\widetilde{b}(t)}{dt} - h\frac{d\widetilde{b}(t)}{dt} \\
&= \varphi_0(h\mathcal{A})\frac{d\widetilde{\widetilde{b}}(t)}{dt} - \widetilde{\widetilde{b}}(t) - h\frac{d\widetilde{b}(t)}{dt}, \qquad (4.8)
\end{aligned}$$

with the expressions of input vectors recalled from previous sections

$$\begin{aligned}
g(t) &= \mathcal{A}x(t) + \mathcal{C}^{-1}Bu(t), \\
\widetilde{g}(t) &= x(t) - \mathcal{G}^{-1}Bu(t), \\
\frac{db(t)}{dt} &= \mathcal{C}^{-1}B\frac{du}{dt}, \\
\frac{d\widetilde{b}(t)}{dt} &= -\mathcal{G}^{-1}B\frac{du}{dt}, \\
\frac{\widetilde{\widetilde{b}}(t)}{dt} &= \mathcal{G}^{-1}\mathcal{C}\mathcal{G}^{-1}B\frac{du}{dt}.
\end{aligned}$$

Therefore, the MEVP term1 can be calculated with either φ_1 or φ_0 function. The MEVP term2 can be calculated with φ_2, φ_1 or φ_0 function. After the stable rational Krylov

subspace is constructed by Arnoldi in Algorithm 4, we adopt the augmented method on the Hessenberg matrix H_m and calculate the MEVPs with different φ_s functions. Since the formulation of MEVPs is modified, the residual $r(h, m)$ used in the Arnoldi process need to be changed. For the formulation with φ_1 function which is in the format of $h\varphi_1(h\mathcal{A})v$, the residual follows

$$r(h, m, \varphi_1) = h\beta h_{m+1,m}(\mathcal{G} + \frac{\mathcal{C}}{\gamma})v_{m+1}e_m^\top H_m^{-1}\varphi_1(h\frac{I - H_m^{-1}}{\gamma})e_1. \tag{4.9}$$

For the formulation $h^2\varphi_2(h\mathcal{A})v$, the residual follows

$$r(h, m, \varphi_2) = h^2\beta h_{m+1,m}(\mathcal{G} + \frac{\mathcal{C}}{\gamma})v_{m+1}e_m^\top H_m^{-1}\varphi_2(h\frac{I - H_m^{-1}}{\gamma})e_1. \tag{4.10}$$

4.1.1 Comparison of φ_s Functions on RC and RLC Networks

With symmetric and nonsingular matrices, the RC network in Sec. 3.3.3 is used to characterize the numerical performance of φ_s functions. Same experimental settings are used while the MEVP term2 is evaluated with higher order φ functions. Fig. 3.6 illustrate the numerical error distribution of the MEVP with φ_0 function. Absolute errors of the results with φ_1 and φ_2 functions are computed with reference to the exact solution and plotted in Fig. 4.2 and 4.3. To validate the stability of the Arnoldi algorithm with structured orthogonalization, we also include the absolute error of result via the original Arnoldi.

We plot the eigenvalues $\lambda(-\mathcal{A}^{-1})$ of the RC network in Fig. 4.4 to better understand the performance of φ_s functions. The error reduces quickly with all φ_s functions by increasing the dimension of rational Krylov subspace, and the slope becomes sharper with larger m. Once the error reaches the lower bound, it cannot be significantly reduced with a larger subspace. When h is large compared to the spectrum of $\mathcal{A}^{-1}$, solution calculated

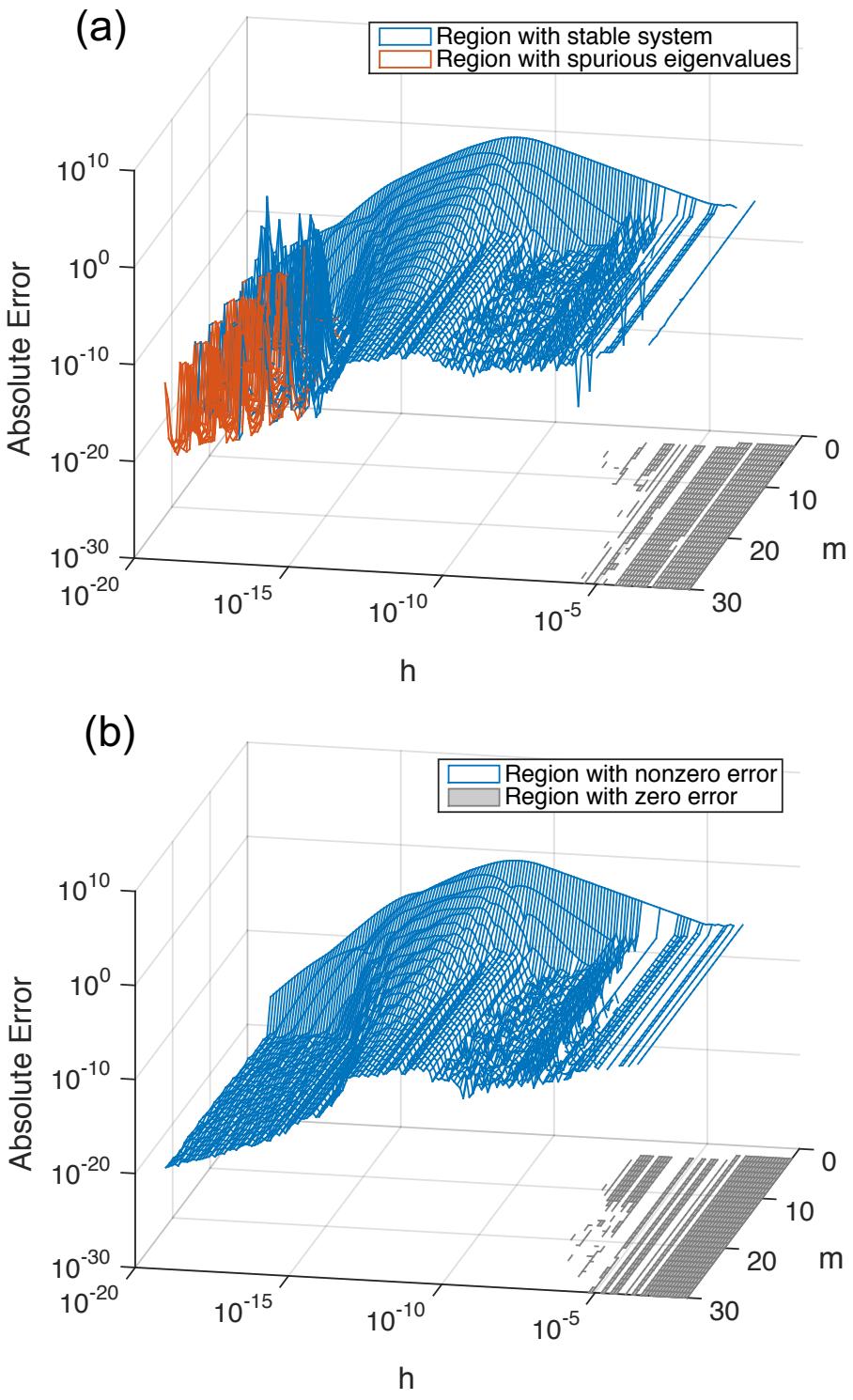

Figure 4.2: RC network with n=100: the absolute error err_{abs} versus h and m is calculated with φ_1 function using (a) original Arnoldi; (b) Lanczos plus structured orthogonalization.

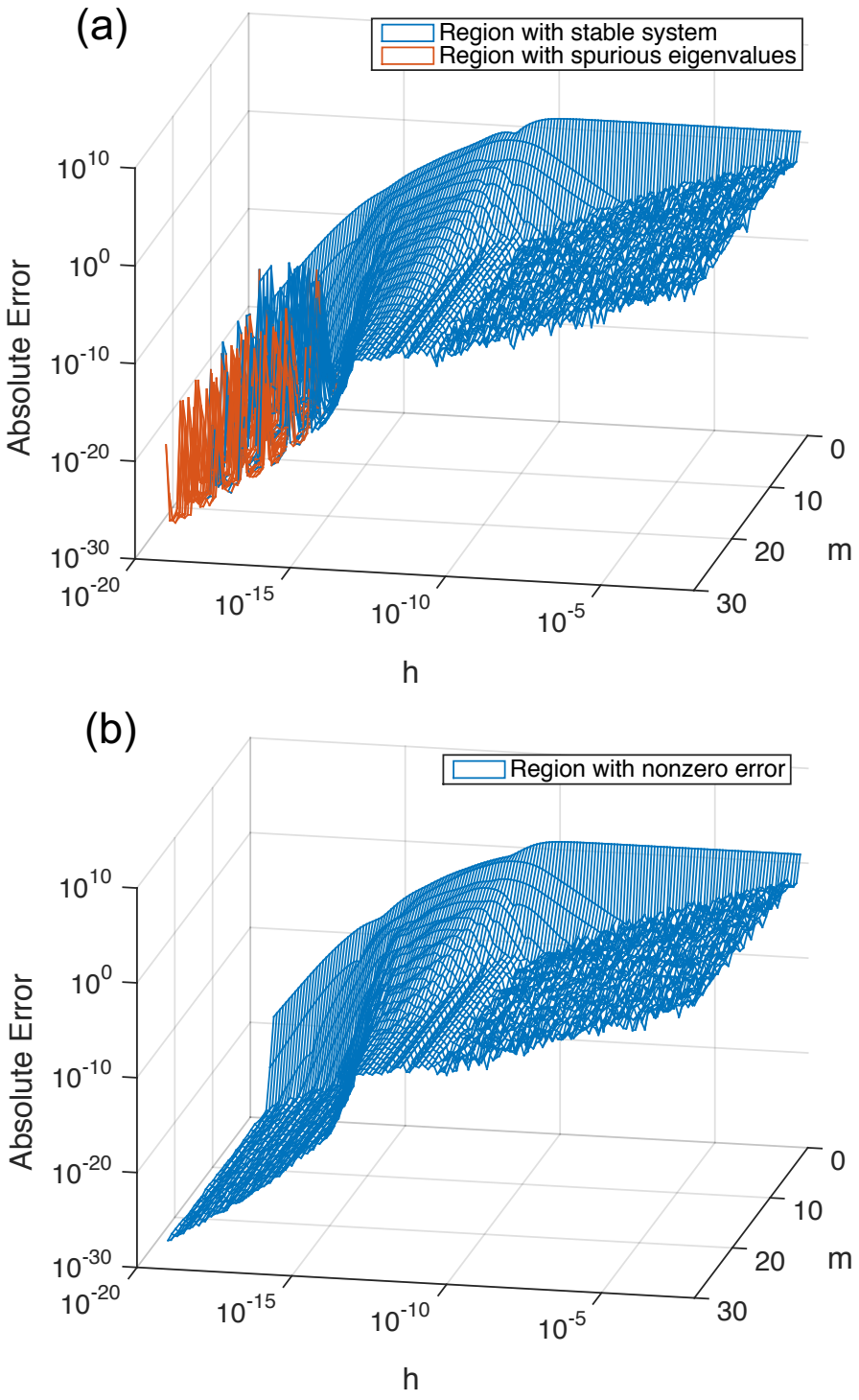

Figure 4.3: RC network with n=100: the absolute error err_{abs} versus h and m is calculated with φ_2 function using (a) original Arnoldi; (b) Lanczos plus structured orthogonalization.

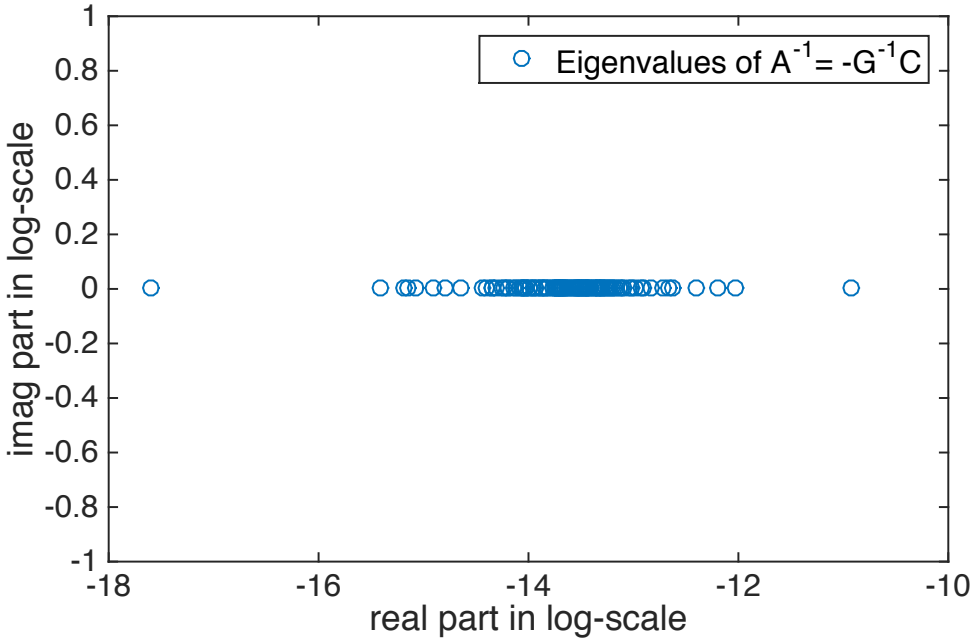

Figure 4.4: RC network with n=100: the eigenvalues of $\mathcal{A}^{-1} = -\mathcal{G}^{-1}\mathcal{C}$ are plotted in log-scale.

with φ_0 function achieves the best accuracy. At the same time, solutions with φ_1 and φ_2 don't show any improvement. If the h is much smaller than the spectrum, the error with φ_1 and φ_2 drops proportional to h.

The behaviors of φ_s functions in each region could be explained with the properties of Krylov subspace methods. Dimension m approximations with Krylov subspace are equivalent to the degree-m polynomial of the exponential term [47]. Here we use v as an abbreviation of the original notation $db(t)/dt$.

- For φ_2 function $h^2\varphi_2(h\mathcal{A})v =$

$$h^2(\frac{I}{2!} + \cdots + \frac{(h\mathcal{A})^m}{(m+2)!} + \mathcal{O}(h^m\mathcal{A}^m))v.$$

- For φ_1 function $h\varphi_1(h\mathcal{A})\widetilde{v} - h\widetilde{v} =$

$$h(\frac{h\mathcal{A}}{2!} + \cdots + \frac{(h\mathcal{A})^m}{(m+1)!} + \mathcal{O}(h^m\mathcal{A}^m))\widetilde{v}.$$

- For φ_0 function $\varphi_0(h\mathcal{A})\widetilde{\widetilde{v}} - \widetilde{\widetilde{v}} - h\widetilde{v} =$

$$(\frac{(h\mathcal{A})^2}{2!} + \cdots + \frac{(h\mathcal{A})^m}{m!} + \mathcal{O}(h^m\mathcal{A}^m))\widetilde{\widetilde{v}}.$$

where the input vectors $v, \widetilde{v}, \widetilde{\widetilde{v}}$ denote the normalized input vectors, which are derived from the relation in Eq 4.8.

Therefore, we can get the following conclusions.

1. When h is small compared to the spectrum of $\mathcal{A}^{-1}$, Taylor expansion of h is valid. The local truncation error is approximated by term $\mathcal{O}(h^m\mathcal{A}^m)$, which is multiplied with h for φ_1 solution and h^2 for φ_2 solution. Knowing that the new vectors constructed could introduce noise and cannot further improve the accuracy if we keep increasing m, the truncation error is dominated by factor of h which explains the floor in Fig. 3.6, 4.2, and 4.3.

2. When h is beyond the spectrum of $\mathcal{A}^{-1}$, Taylor expansion of h^{-1} is valid. By using a large step size the dynamic behavior of RC circuit is dominated by the slow decaying component, the solution is well approximated with a small m.

3. When h lies in the range of the eigenvalues of $\mathcal{A}^{-1}$, increasing m could significantly improve accuracy so the error follows the polynomial term with m.

With given h and approximated eigenvalue spectrum of the system, we could choose the appropriate φ_s function to obtain higher accuracy and improve convergence rate. Different colors in Fig. 4.5 denote the choices of φ_s functions with lowest error.

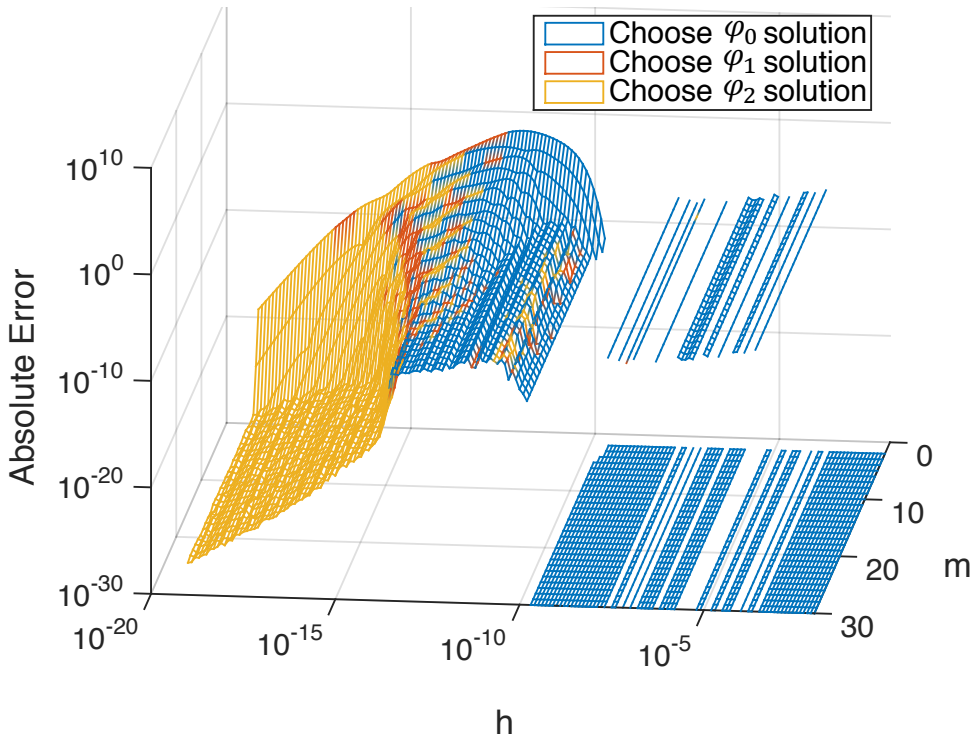

Figure 4.5: RC network with n=100: choices of φ_s functions are plotted with different colors versus h and m. The zero error of φ_0 solution is set to 10^{-30}.

We apply the evaluations of MEVPs with φ_s functions to the RLC network as well. Experiments in Sec. 3.3.3 are operated on the same test case. Fig. 3.7 illustrates the performance of φ_0 function. The absolute errors of φ_1 and φ_2 functions are plotted in Fig. 4.6 and Fig. 4.7, respectively. The limited application of φ functions with singular $\mathcal{C}$ matrix could be resolved with implicit regularization. The original Arnoldi process comes across more severe stability problems with higher order φ_s functions, which are the cause of discontinuities on the figures. The spectrum of the eigenvalues lies in the complex plane with negative real part, as shown in Fig. 4.9. The choices of φ_s functions on the RLC are plotted in Fig. 4.8, which is consistent to the observation on RC network.

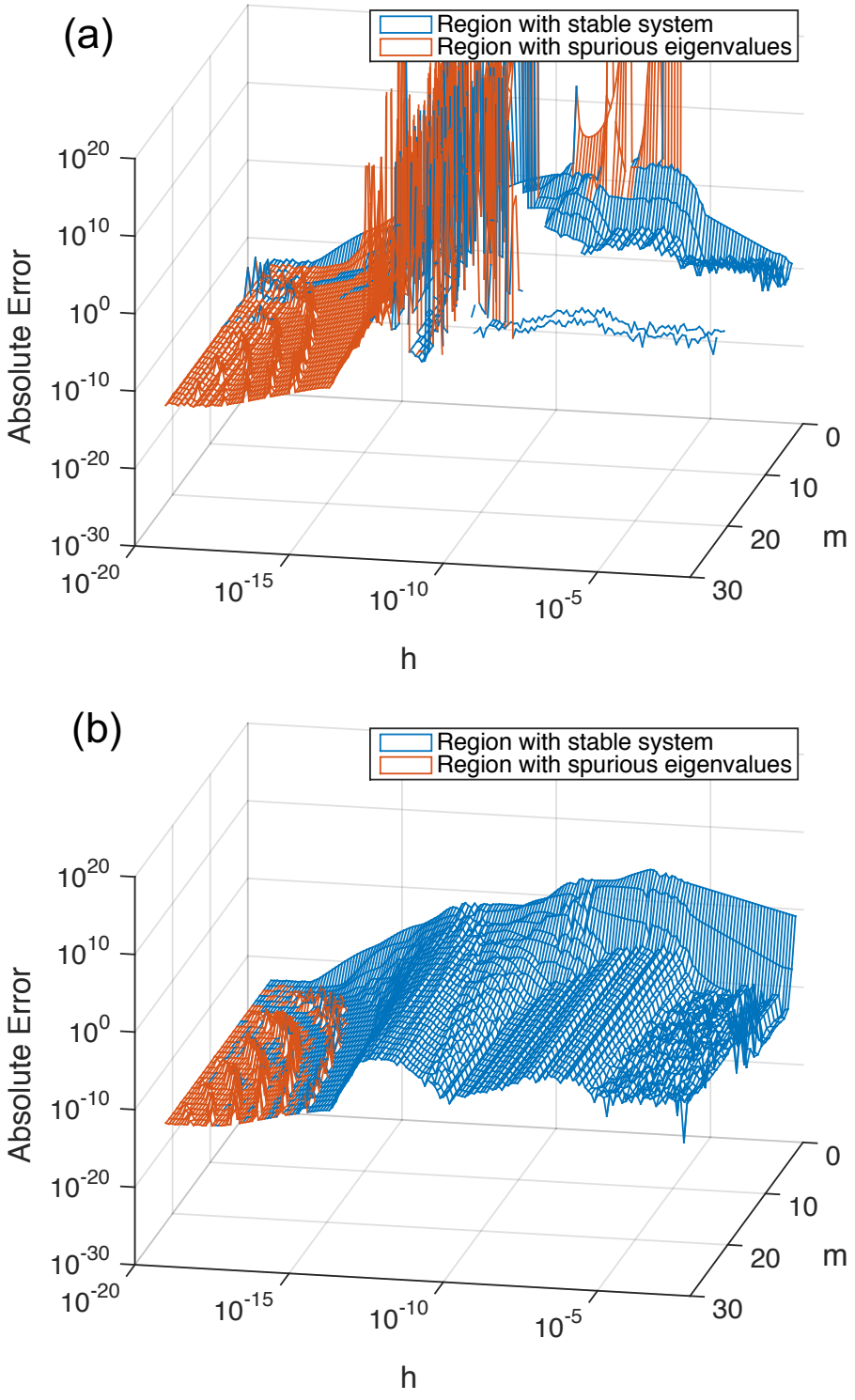

Figure 4.6: RLC network with n=507: the absolute error err_{abs} versus h and m is calculated with φ_1 function using (a) original Arnoldi; (b) Arnoldi plus structured orthogonalization. The reference solution is from the explicit calculation of underlying ODEs.

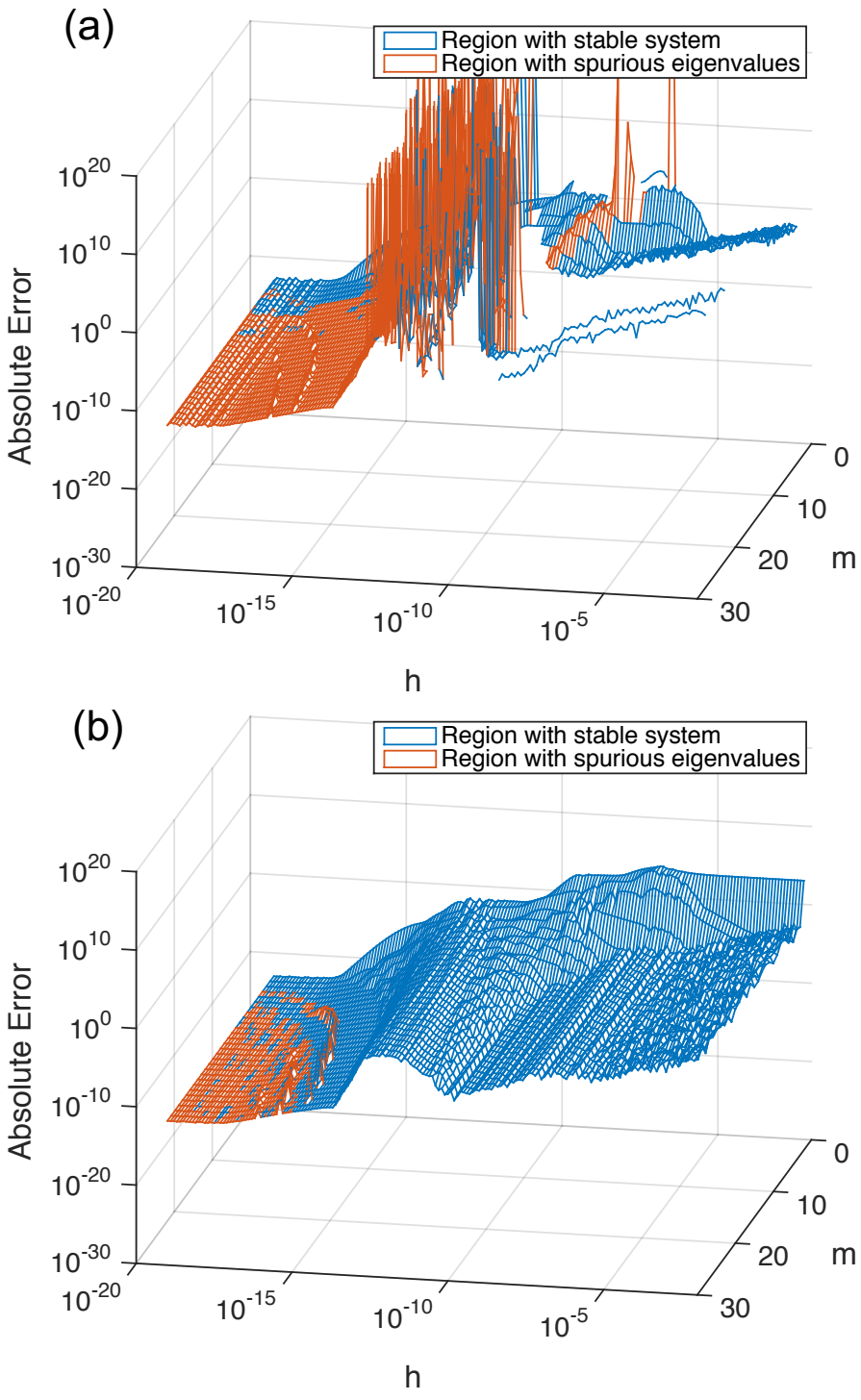

Figure 4.7: RLC network with n=507: the absolute error err_{abs} versus h and m is calculated with φ_2 function using (a) original Arnoldi; (b) Arnoldi plus structured orthogonalization. The reference solution is from the explicit calculation of underlying ODEs.

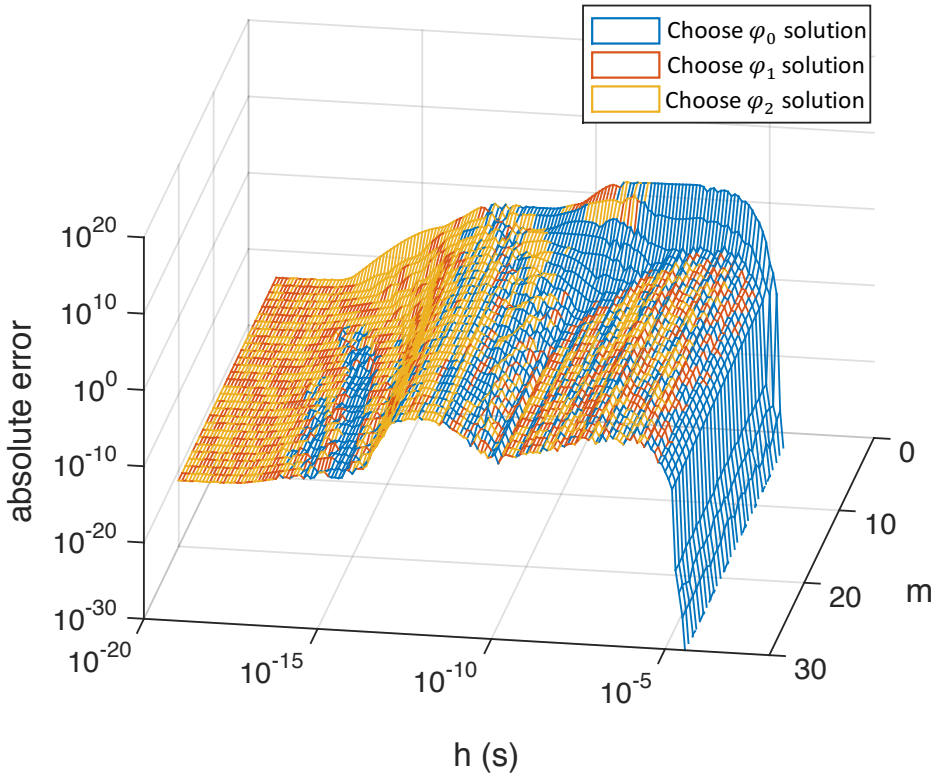

Figure 4.8: RLC network with n=507: choices of φ_s functions are plotted with different colors versus h and m.

4.2 Exploration of the Local Optimal Ratio in Rational Krylov Spectral Transformation

The convergence rate of rational Krylov subspace is closely related to the spectrum of system and could be improved by choosing appropriate ratio γ to confine the spectrum. The dominant eigenvalues to characterize the dynamical behaviors of circuits are found in the first several iterations of Arnoldi process. It is proved that the optimal choice of γ is inversely proportional to the dimension of Krylov subspace [53]. A step size h is given in the matrix exponential as a coefficient of recovered system matrix, the γ is empirically set at the same scale of h.

Lemma 4.2.1. The local optimal result of computing MEVPs with rational Krylov subspace

is achieved by choosing $\gamma \approx \frac{h}{m}$, where h is the given step size and m is the dimension of Krylov subspace.

The RLC network in Sec. 3.3.3 is used to validate the statement. We adopt the Arnoldi algorithm with structured orthogonalization and implicit regularization to calculate the MEVPs. To preclude the different numerical performances of φ_s functions, we only calculate the MEVPs in expression of φ_0. The target function is as follows

$$f(h, m, \gamma) = e^{h\mathcal{A}}v \approx \beta V_m e^{h\frac{I-H_m^{-1}}{\gamma}} e_1. \tag{4.11}$$

The distribution of eigenvalues is plotted in Fig. 4.9 and four sample points of h are selected. We increase the dimension m of Krylov subspace from 2 to 30 and set γ proportional to h with different ratios. For each given h and m, the optimal γ with respect to m is found by computing the absolute error of $f(h, m, \gamma)$

$$\gamma_{opt}(h, m) \Leftarrow \min_\gamma ||f(h, m, \gamma) - f_{ref}(h)||, \tag{4.12}$$

where $f_{ref}(h)$ is the reference solution by solving the underlying ODEs explicitly. Fig. 4.10 plots the ratio between γ_{opt} and h versus m at each selected h. The relation in Lemma 4.2.1 is observed when h is close to the majority of the eigenvalues of $\mathcal{A}^{-1}$. When $h = 3 \times 10^{-13}s$, the eigenvalues corresponding to this range are not well separated so that a larger dimension of Kyrlov subspace could significantly improve the results. Similar phenomenon can be found for $h = 10^{-11}s$, which converges faster since there are less dominant eigenvalues in this range. In other cases when h is close to the borders of eigenvalues, solution can converge at a small m. There is not much potential of improvements by setting an appropriate γ. The total error distributions versus m and $\frac{h}{\gamma}$ of the selected step sizes are plotted in Fig. 4.11, where the minimum error at γ_{opt} is plotted with red marker. The relative error is

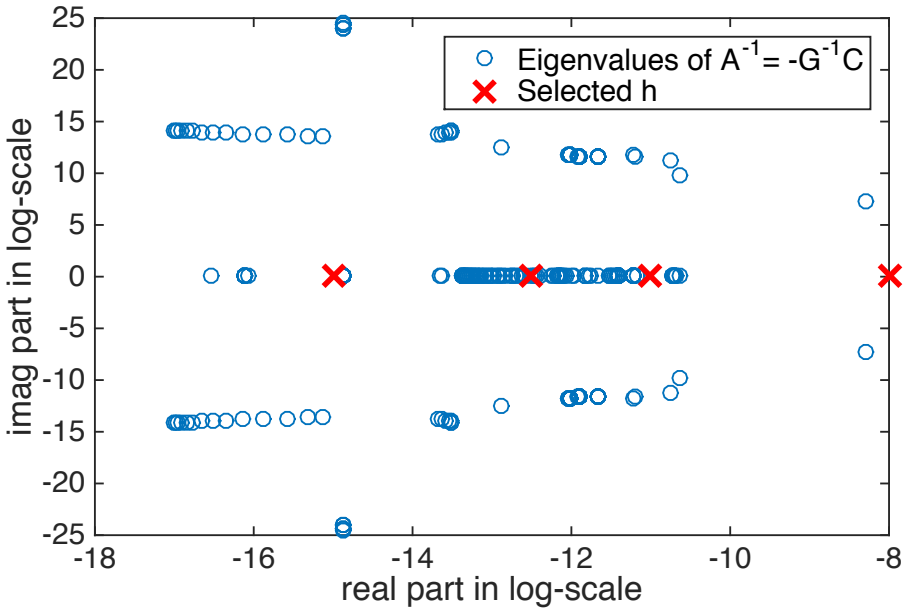

Figure 4.9: RLC network with n=507: the eigenvalues of $\mathcal{A}^{-1} = -\mathcal{G}^{-1}\mathcal{C}$ in log-scale. Four different step sizes set in MEVPs are compared to the spectrum.

calculated with

$$err_{rel} = ||f(h, m, \gamma) - f_{ref}(h)||/||f_{ref}(h)||. \tag{4.13}$$

Once the error reaches the floor (flat region), which means numerical rounding error becomes dominant, a larger m doesn't help to improve the accuracy. Then the accuracy of results no longer relies on the choice of γ.

4.3 Total Simulation Framework

The proposed techniques are integrated to the multi-step integration framework for PDN transient simulation, as shown in Figure 4.12. The DAEs in Eq. 2.2 are constructed via **MNA**. The whole simulation time is discretized into the time grids $T = [0, t_1, t_2, \ldots, t_n]$ depending on the breakpoints of input sources, which provide the maximum allowable step sizes for circuit transient simulations. The time interval between any two time points is

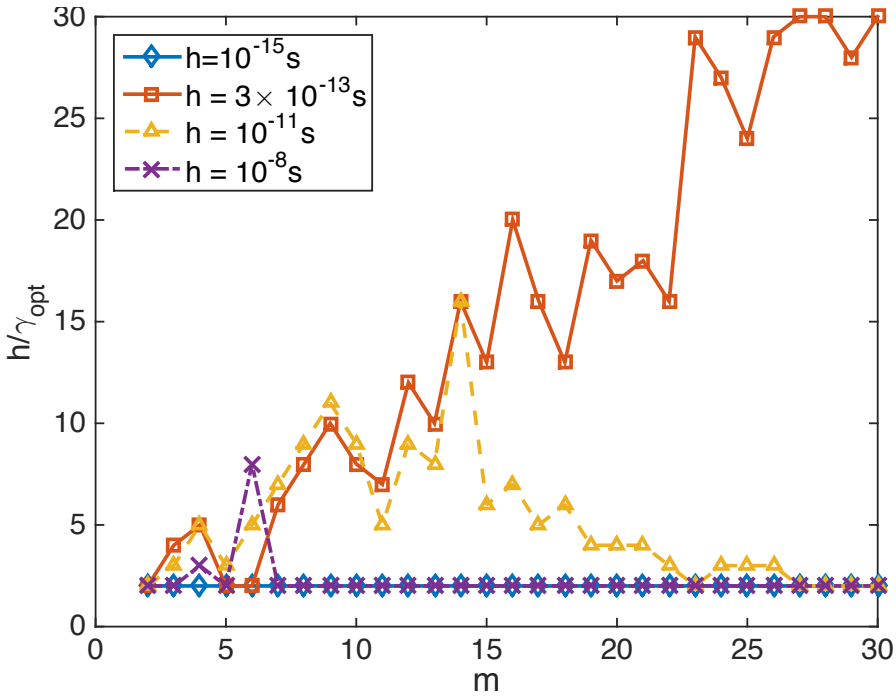

Figure 4.10: RLC network with n=507: the ratio h/γ_{opt} with minimum relative error at each m is plotted.

computed with $h = t_{k+1} - t_k$. Flexible steps could be used to simulate the intermediate state between any two time points of interest, in our framework the maximum stepping scheme is adopted to speed up the analysis. The transition activities of PDNs' load models are characterized by the pulse/PWL input sources. The different step sizes are saved in the set $S_h = \{h_1, h_2, \ldots, h_s\}$, where s is usually much smaller than the total number of time grids n or input sources.

In the **Matrix Preparation** step, LU decomposition is performed on matrices $\mathcal{G}$ and $\gamma\mathcal{G} + \mathcal{C}$. The results will be used in the DC and transient analysis. The performance of MEVPs is affected by the spectrum of system and the current step size h. We adopt the rational Krylov subspace method and choose γ to shift the original system in order to capture the dominant eigenvalues. Empirically γ is set proportional to h [6] and results could be optimized with the best ratio. We define a group of initial ratio in S_γ for each different $h_l \in S_h, l = 1, 2, \ldots$.

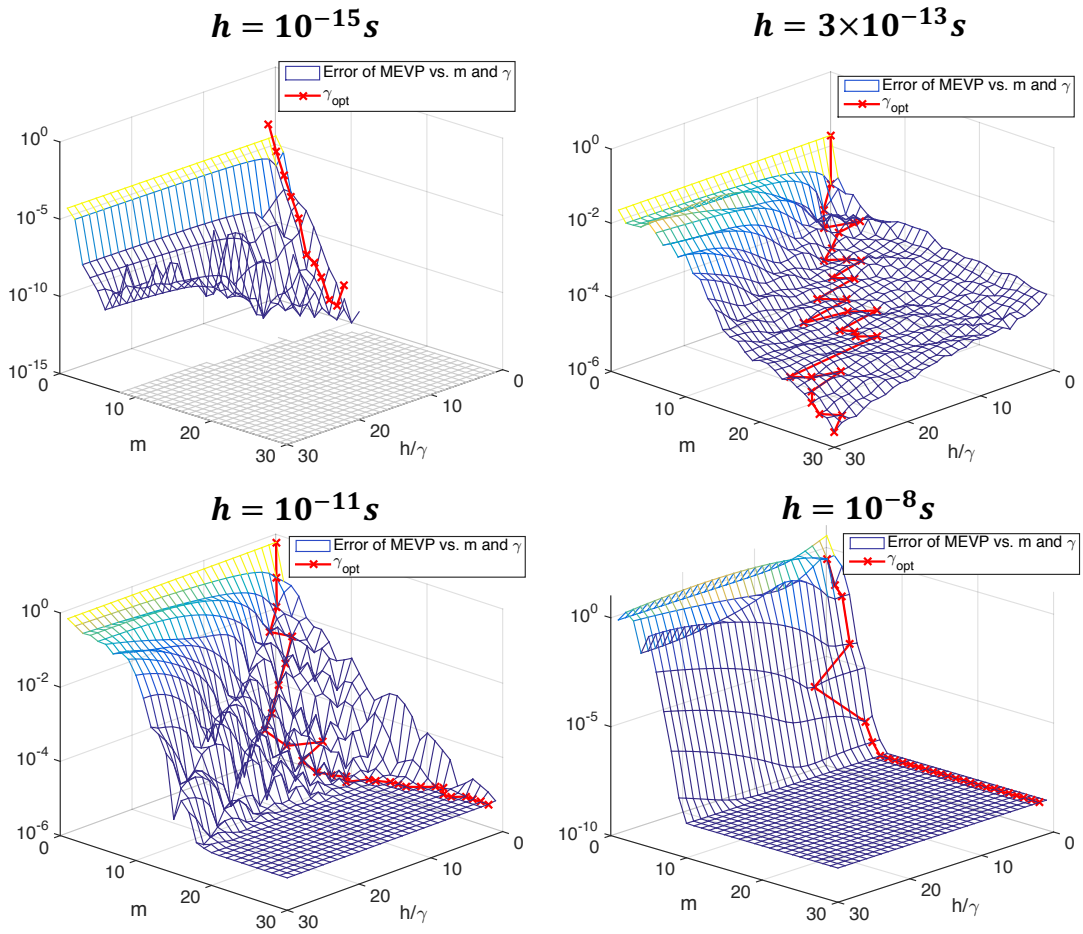

Figure 4.11: RLC network with n=507: relative error of MEVP computations err_{rel} versus m and γ. The local optimal γ_{opt} is denoted with red marker.

DC Analysis provides the initial solution $x(0)$ for transient simulation where the linear system is directly solved with $LU(\mathcal{G})$. A single integration step in transient analysis is described afterwards. In step **MEVP Settings**, the devised techniques are implemented as follows,

- Set φ functions for each MEVP term. The step size h is read from the time grids in T. Together with the approximated spectrum of system, h is used to determine the appropriate φ function for the evaluation of MEVPs.

- Set $\gamma \approx \gamma_{opt} \in S_\gamma$ and reuse the results of $LU(\gamma_{opt}\mathcal{G}+\mathcal{C})$ from the Matrix Preparations

step.

Sec. 4.1 demonstrates the performance of φ functions. The decomposed $(\gamma\mathcal{G}+\mathcal{C})$ matrices are reused with the optimal approximated γ_{opt} for the construction of rational Krylov subspace. Details of the optimal ratio are covered by Sec. 4.2.

Algorithm 4 implements the Arnoldi with structured orthogonalization to evaluate the MEVPs with stability preserved results. The key operations include

- replacing all the normalization steps by a semi-inner product with $\mathcal{C}$,
- applying implicit regularization if $\mathcal{C}$ is singular.

When the solution of current step satisfies the convergence check, the process proceeds to the next time step until the whole time span is covered. Consider some corner cases with ill-conditioned systems, a **Numerical Pruning** process is performed on the approximation of MEVPs.

4.4 Simulation Results

The simulation framework is implemented in MATLAB and uses UMFPACK package for LU decomposition. We perform the experiments on a Linux server with Interl(R) Xeon(R) CPU E5-2640 v3 2.60GHz and 125 GB memory.

4.4.1 Leap Function for Multiple Frequencies

We use the off-chip PDN with board, package and chip [48] consisting multiple resonant frequencies for transient simulation. The schematic circuit is shown in Fig. 4.13 as a three tanks RLC model. Usually for the off-chip PDN with high stiffness, it is hard to capture all the oscillation components in the transient simulation. The high order polynomial approximation with Krylov subspace of the MEVPs breaks the limitation of

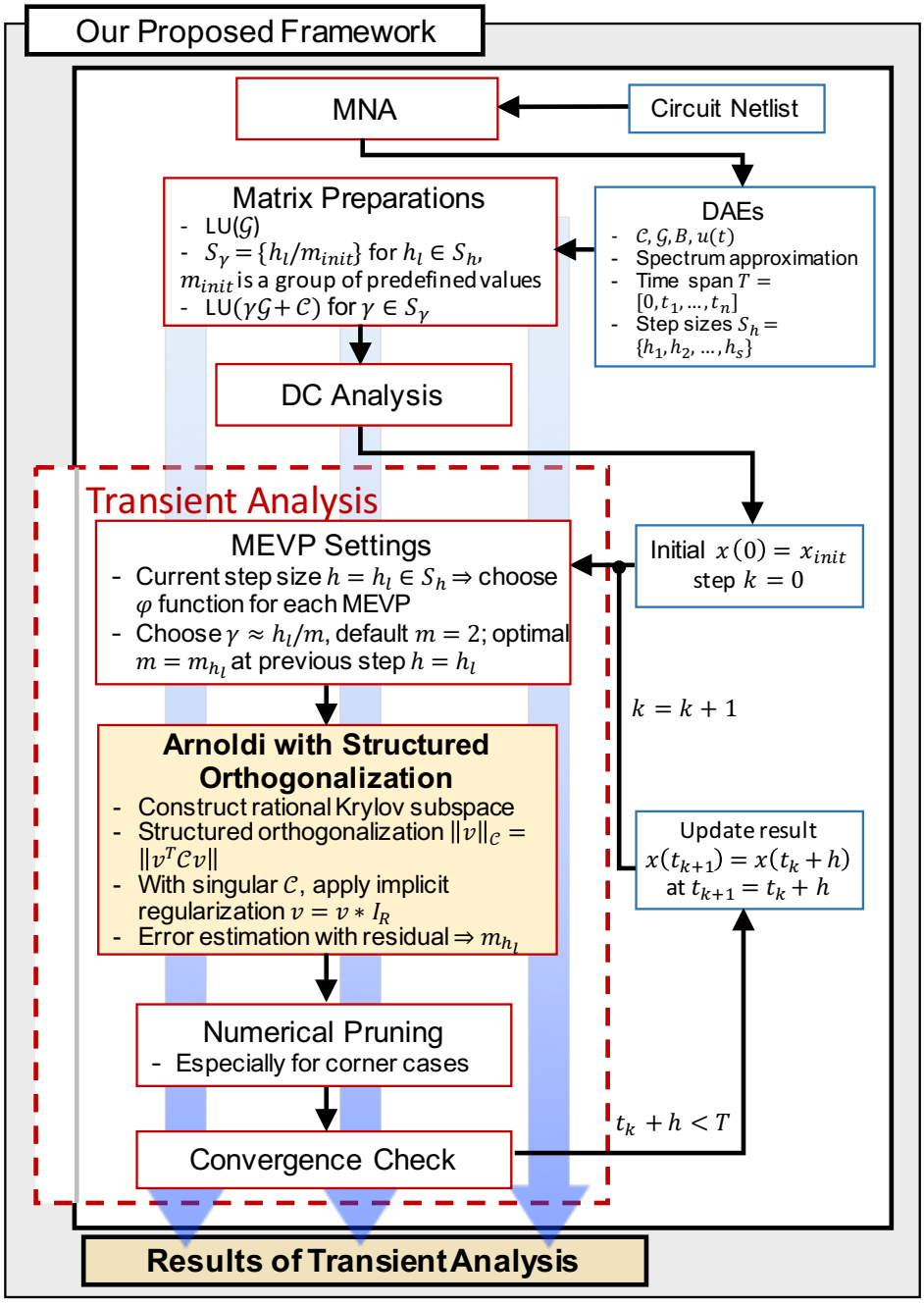

Figure 4.12: Total framework of PDN transient simulations with matrix exponential based integration method.

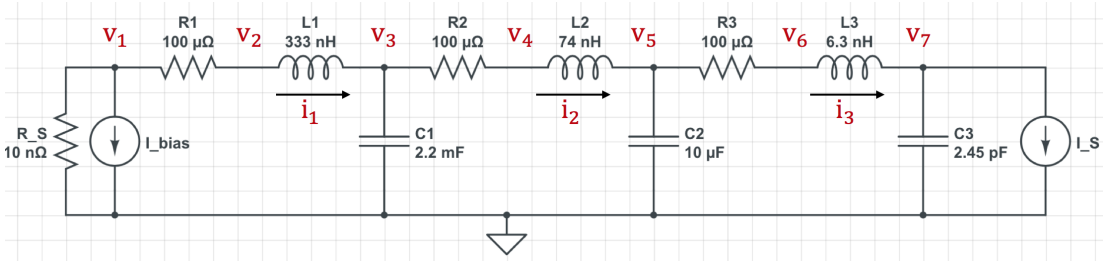

Figure 4.13: Off-chip PDN with $R1 = 100\mu\Omega, L1 = 333nH, C1 = 2.2mF, R2 = 100\mu\Omega, L2 = 74pH, C2 = 10\mu F, R3 = 100\mu\Omega, L3 = 6.3nH, C3 = 2.45pF$.

Dahlquist barrier [8, 57, 64], and enables the simulation with adaptive step sizes while obtaining the accurate results. After applying the structured orthogonalization and implicit regularization in the Arnoldi algorithm, stable Krylov subspace is generated. An adaptive stepping scheme is implemented in our framework. In order to capture the oscillation curve with low frequency and finish the simulation in a short time, a large step size is chosen to skip the higher frequency components. The details between any two time points could be computed with a smaller step size. MEVPs with Krylov subspace methods provide the flexible choice of step sizes and accurate results.

The bias I_{bias} is set to zero. The PWL input I_S start to increase from zero at $t = 0$ with rising time TR= $1ps$ and keeps stable in the following stage. We apply the simulation framework in Fig. 4.12 to calculate the transient response of the off-chip PDN. Adaptive step sizes are used in different simulation stages to capture the dominant resonant frequencies. Exact solution is calculated as reference for the whole system by PWL response from MATLAB continuous-time transfer function. To observe all the frequency components, the simulation time $T = 1.5ms$ is taken. Fig. 4.14 shows the simulation results on node voltage v_5, where different step sizes are used to approximate the oscillation waveforms of certain frequencies.

- In Fig. 4.14 (a), the low frequency component $f_l = 5.88kHz$ is reflected in a long simulation time $T = 1.5ms$. Large step size $h = 1.8 \times 10^{-5}s$ is chosen to capture the

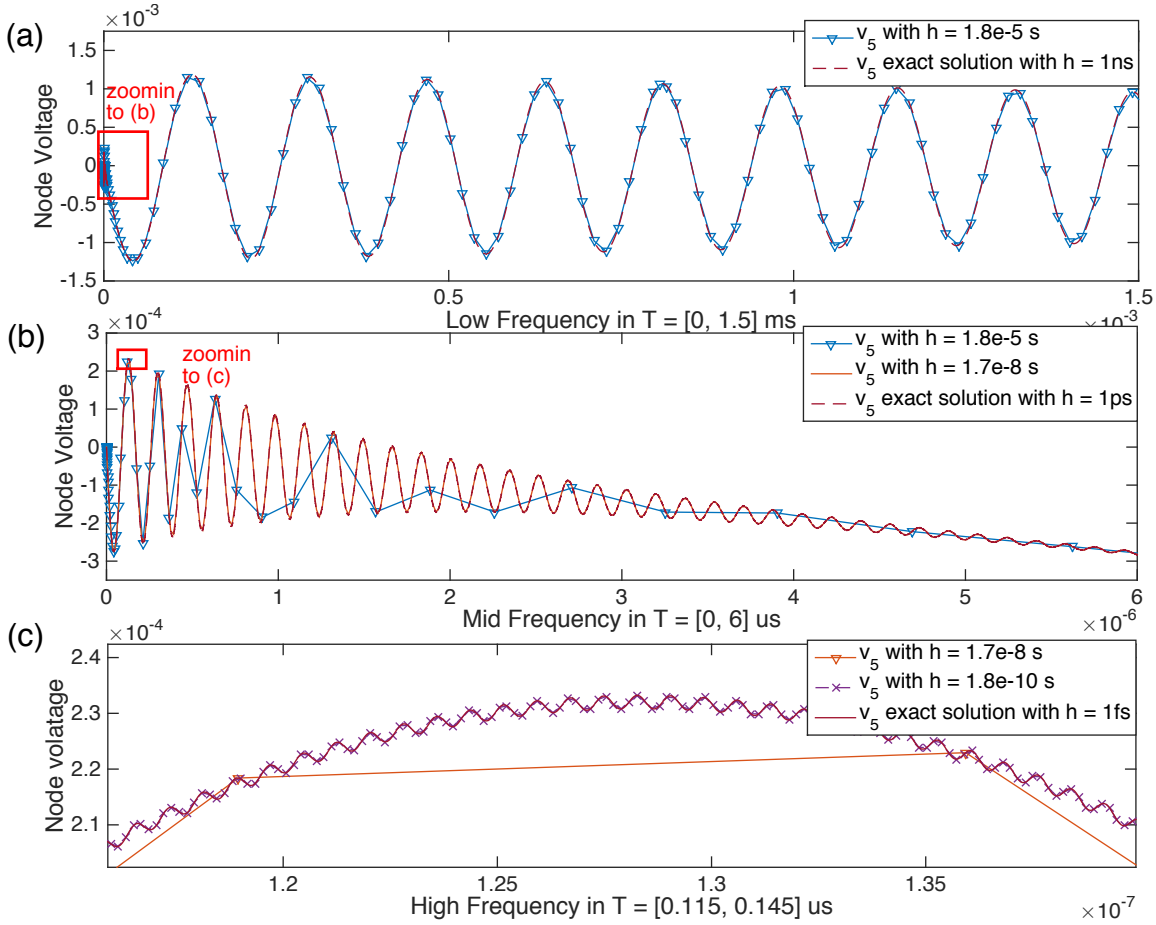

Figure 4.14: Adaptive step sizes are applied for multiple frequency components of off-chip PDN in different simulation stages.

low frequency efficiently and skip the higher frequency components.

- In Fig. 4.14 (b), the medium frequency component $f_m = 5.86MHz$ decays in the early simulation stage $T = 6us$. A relatively smaller step $h = 1.7 \times 10^{-8}s$ is used to capture the oscillations. The results calculated with larger step sizes are included to show the adaptive stepping scheme.

- In Fig. 4.14 (c), the high frequency component $f_h = 1.28GHz$ could be accurately captured with a even smaller step $h = 1.8 \times 10^{-10}s$, which cannot be observed with the larger step sizes.

Results show consistence to exact solution with different step sizes, showing that our results could achieve high accuracy. The maximum absolute error is $0.9\mu V$ for result with $h = 1.8 \times 10^{-5}s$, $6.04nV$ for $h = 1.7 \times 10^{-8}s$ and $2.49nV$ for $h = 1.8 \times 10^{-10}s$. In the earliest simulation stage when the input is not stable, the step sizes are limited by the breakpoints of input sources in all the simulations. In the following stable stage, the step sizes grow gradually until they reach the maximum step limitation.

4.4.2 System-Level PDN Transient Simulations with φ Functions

The system-level PDNs are constructed by connecting the on-chip PDNs to the off-chip system, which is modeled as the three tank RLC in Fig. 4.13. We adopt the IBM power grid benchmarks [38] as well as general 3D PDNs. Table 4.2 displays the design specifications of the test cases of which size ranges from 45K to 7M. For the ibmpg1t to ibmpg6t, the time grids are determined by interleaving the breakpoints of input sources. We adopt the maximum allowable step sizes between any two time grids. Most of the IBM power grids have 141 breakpoints, so 140 integration steps are performed in the transient simulation. The total simulation time interval is divided into 44 steps for ibmpg4t. The general PDNs contain more inductors with only one supply voltage source and one current source. The simulation of pdn1 to pdn4 includes 10 pulse cycles of the input current source with 40 steps.

We operate transient simulations on the system-level PDNs with our proposed framework and advanced techniques. According to the adaptive stepping scheme, we could have the step sizes at distinct values. The time intervals in ibmpg1t to ibmpg6t vary from $10ps$ to $0.59ns$. In the pdn1 to pdn4, the step sizes are depending on the rising (TR= $100ps$) and falling (TF= $100ps$) time, the pulse width (PW= $200ps$), and the period (T= $0.1ms$). As described in Sec. 4.1, the numerical performance of matrix exponential is affected by the spectrum of system and current step size, which can be improved by

Table 4.2: Design Specifications of PDN Cases.

Design	#Node	#R	#C	#L	#I	#V
ibmpg1t	54K	41K	11K	277	11K	14K
ibmpg2t	165K	245K	37K	330	37K	330
ibmpg3t	1.0M	1.6M	201K	955	201K	955
ibmpg4t	1.2M	1.8M	266K	962	266K	962
ibmpg5t	2.1M	1.6M	473K	277	473K	539K
ibmpg6t	3.2M	2.4M	761K	381	761K	836K
pdn1	45.7K	23K	15K	15K	1	1
pdn2	688K	349K	229K	229K	1	1
pdn3	2.9M	1.47M	965K	965K	1	1
pdn4	7.4M	3.75M	2.47M	2.47M	1	1

using appropriate φ_s functions. Knowing the system spectrum and step sizes in advance, we are able to determine the range of MEVPs and choose an appropriate φ_s function. In Table 4.3, the spectrum of $\mathcal{G}^{-1}\mathcal{C}$ is approximated by measuring the dominant eigenvalues. We provided the choice of φ_s functions for each MEVP term with certain step size. The expressions of MEVPs are given in Eq. 4.8.

Table 4.3: Application of Optimal φ Functions to the MEVPs of PDNs.

Design Group	Approx. spectrum	Step Sizes (s)	MEVP	Choices of φ
ibmpg 1t to 6t	2×10^{-9}	$[1, 4, 5] \times 10^{-11}$	term1	φ_1
			term2	φ_2
		5.90×10^{-10}	term1	φ_0 or φ_1
			term2	φ_1 or φ_2
ibmpg$_{fast}$ 1t to 6t	2×10^{-9}	1×10^{-13} 4.99×10^{-11}	term1	φ_1
			term2	φ_2
		5.99×10^{-10}	term1	φ_0 or φ_1
			term2	φ_1 or φ_2
pdn 1 to 4	$\left[10^{-7}, 10^{-16}\right]$	$[1, 2] \times 10^{-10}$	term1	φ_0
			term2	φ_1
		1×10^{-3}	term1	φ_0
			term2	φ_0
pdn$_{mid}$ 1 to 4	$\left[10^{-7}, 10^{-16}\right]$	$[0.5, 1] \times 10^{-11}$	term1	φ_0
			term2	φ_0 or φ_1
		1×10^{-3}	term1	φ_0
			term2	φ_0
pdn$_{fast}$ 1 to 4	$\left[10^{-7}, 10^{-16}\right]$	$[1, 2] \times 10^{-13}$	term1	φ_0 or φ_1
			term2	φ_1 or φ_2
		1×10^{-3}	term1	φ_0
			term2	φ_0

To further explore the performance of our framework, we modify the input source

waveforms to provide different time grids. The transition time of input sources shrinks to different scales while the total simulation time remains unchanged. The eigenvalues of IBM power grids are mostly close to a certain value. We shrink the transition time of the pulse waveforms from $10ps$ to $0.1ps$ and denote the new test cases with subscript *fast*. The general PDNs are very stiff with more oscillation components, experiments are designed by setting the transition time of inputs in range $[10^{-13}, 10^{-10}]$. The new test cases are denoted by pdn_{mid} and pdn_{fast} to describe the response to input.

In the framework describe by Fig. 4.12, the LU decomposition of $\mathcal{G}$ and $(\gamma\mathcal{G} + \mathcal{C})$ is performed prior to the transient simulation, where the ratio γ is selected according to the study in Sec. 4.2. The optimal setting γ_{opt} is around h/m when h lies in the range of $\mathcal{G}^{-1}\mathcal{C}$. After parsing the input sources (with PWL waveforms) and get the time grids, a set of m_{init} are chosen to approximate the dimension of Krylov subspace. The optimal ratio $\gamma = h_l/m_{init}$ is calculated for each possible step size h_l. In the first integration, we use the default $\gamma = h/2$ and save the converged dimension m_h as criteria for later calculations. Actually, there only exist a few distinct step sizes so that the computation cost is affordable and worthy for simulation efficiency. In our simulation framework, we choose $m_{init} = \{2, 4, 8, 16\}$ empirically and update the optimal ratio as

$$\gamma_{opt} = h/m_{init} \Leftarrow \min_{m_{init} \leq m_h} m_h - m_{init}.$$

The simulation performance is summarized in Table. 4.4. The dimensions of Krylov subspaces for MEVPs are marked as m_1 and m_2. The average is computed among all the nonzero values. The maximum dimension is included as m_{peak}. Experiments using the default ratio $\gamma = h/2$ and optimal ratio γ_{opt} are performed to show the improvement on convergence rate. Simulation results are compared to the reference solution with TRAP method, which adopts much smaller step sizes. The relative difference of solution is

computed with $||x_{mevp} - x_{ref}||_\infty / ||x_{ref}||_2$. Since same tolerances are used for the convergence of Arnoldi process, the results with default ratio and optimal ratio achieve similar accuracy. We only provide the relative difference with optimal ratio in Table 4.4.

For the ibmpg1t to ibmpg6t with a narrow spectrum, the calculation of MEVPs converges faster and a smaller Krylov subspace is used to generate accurate results. The improvement with optimal ratio is slight. With a relatively large step size like 1ms, the Arnoldi process could converge faster because only the smallest frequency is dominant to characterize the response. The improvement in the stiff pdn cases is significant with optimal ratio, where a larger Krylov subspace is required in the simulation to characterize the dynamical behaviors. The fastest transition time lies on the border of the system's spectrum, the computation of MEVPs converges very fast and the default ratio provides the optimal results. The transient simulations of pdn_{fast} are completed in a shorter time. The performance of pdn_{mid} is in between. Simulation results on the stiff PDNs are consistent to the observations in Sec. 4.2.

4.5 Summary

We further explore the exponential related φ functions which enable the multiple computation methods for MEVPs. With given system spectrum and step sizes, an appropriate choice of φ function could resolve the potential numerical error and increase the accuracy. We also figure out the optimal ratio in rational Krylov subspace, which is used to confine the spectrum of original system and helps to capture the dominant eigenvalues more efficiently. The techniques are integrated in our simulation framework and applied to the PDN simulation tasks.

For system-level PDNs, adaptive stepping scheme is adopted to speed up the transient simulation. Stable results with Krylov subspace are generated through the Arnoldi algorithm

with structured orthogonalization and implicit regularization. Appropriate choices of φ functions and ratio γ improve the convergence of MEVPs. Results with maximum allowable step sizes show high accuracy, showing that our matrix exponential based integration method could achieve efficient and accurate performance for transient simulation of large-scale PDNs. The application to general large dynamical systems are worth exploring for further improvement.

Chapter 4 is a reprint of the material as it appears in the following two works: P. Chen, C. K. Cheng, D. Park, and X. Wang, "Transient circuit simulation for differential algebraic systems using matrix exponential," in *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018. X. Wang, P. Chen, and C.-K. Cheng, "Stability and convergency exploration of matrix exponential integration on power delivery network transient simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019. The author is the primary author and investigator of the papers.

Table 4.4: Transient Simulation Performance of PDN cases. Tran(s): runtime of transient simulation excluding the DC analysis; $m_{1,avg}$ and $m_{2,avg}$: average dimension m for each MEVP term; $m_{1,peak}$ and $m_{2,peak}$: maximum dimension m for each MEVP term; Max Diff(%): maximum relative difference in percentage compared to reference solution.

	Default Ratio					Optimal Ratio					
Design	Tran(s)	$m_{1,avg}$	$m_{2,avg}$	$m_{1,peak}$	$m_{2,peak}$	Tran(s)	$m_{1,avg}$	$m_{2,avg}$	$m_{1,peak}$	$m_{2,peak}$	Max Diff(%)
ibmpg1t	42.9	2.39	2.62	7	3	41.9	2.27	2.62	5	3	2.64×10^{-5}
ibmpg2t	179.3	2.36	2.62	7	3	168.8	2.23	2.55	5	3	3.55×10^{-5}
ibmpg3t	1675.6	2.34	2.46	7	3	1675.3	2.24	2.46	6	3	8.66×10^{-5}
ibmpg4t	753.7	3.63	2.46	9	3	727.5	3.24	2.46	9	3	1.99×10^{-4}
ibmpg5t	2366.0	2.36	2.46	7	3	2337.0	2.23	2.46	5	3	4.27×10^{-5}
ibmpg6t	3744.0	2.32	2.46	7	3	3709.6	2.23	2.46	5	3	2.66×10^{-5}
ibmpg1t$_{fast}$	44.5	2.82	2.62	7	3	37.9	2.64	2.62	5	3	2.35×10^{-5}
ibmpg2t$_{fast}$	173.3	2.34	2.62	7	3	171.5	2.21	2.62	5	3	5.47×10^{-5}
ibmpg3t$_{fast}$	1798.7	2.36	2.59	7	3	1770.9	2.25	2.59	6	3	6.99×10^{-5}
ibmpg4t$_{fast}$	994.2	3.49	2.54	9	3	829.5	3.10	2.54	9	3	4.06×10^{-4}
ibmpg5t$_{fast}$	2408.0	2.45	2.46	7	3	2337.1	2.32	2.46	5	3	5.03×10^{-5}
ibmpg6t$_{fast}$	3682.7	2.36	2.46	7	3	3681.2	2.23	2.46	5	3	4.33×10^{-5}
pdn1	27.27	9.67	17	19	17	16.58	6.70	11.30	12	11	2.08×10^{-4}
pdn2	699.9	9.67	17	19	21	430.5	6.70	11.30	12	11	2.25×10^{-4}
pdn3	2714.6	9.67	13	19	13	2342.8	6.70	12.05	12	12	2.63×10^{-4}
pdn4	11635	11.33	17	24	17	9155.4	8.10	13.25	14	13	1.59×10^{-4}
pdn1$_{mid}$	14.63	6.67	9	10	9	12.92	5.50	8.05	9	8	5.06×10^{-4}
pdn2$_{mid}$	322.6	7	9	11	9	266.5	5.53	8.05	9	8	4.84×10^{-4}
pdn3$_{mid}$	1865.2	7.73	9	13	9	1342.4	5.53	8.05	9	8	4.76×10^{-4}
pdn4$_{mid}$	6457.3	8	9	14	9	5489.0	6.63	9	10	9	4.66×10^{-4}
pdn1$_{fast}$	8.67	2.67	5	4	5	Same to Default Ratio					1.95×10^{-3}
pdn2$_{fast}$	144.8	2.67	4	4	4	Same to Default Ratio					1.25×10^{-2}
pdn3$_{fast}$	765.0	2.67	4	4	4	Same to Default Ratio					1.09×10^{-2}
pdn4$_{fast}$	2798.2	2.67	4	4	4	Same to Default Ratio					1.22×10^{-2}

Chapter 5

Novel Integration Algorithms for Nonlinear Circuit Simulation

5.1 Motivation

With the existence of nonlinear elements, the circuits are formulated as nonlinear equations. The conventional implicit integration methods, such as BE and TRAP, are preferred because of the stability in the transient simulation. Newton-Raphson iterations are applied to obtain the converged solution at each time step. In each iteration, the nonlinear elements are linearized and the linear system is solved [8, 34, 37]. Therefore, the implicit integration methods are more computationally expensive compared to the explicit integration methods. However, the results of conventional methods have limited accuracy. To ensure the accuracy while improving the convergence rate, we devise the explicit matrix exponential based integration for the nonlinear circuit transient simulation.

In Sec. 5.2, we apply the matrix exponential based integration method to the transient simulation of analog designs. The contributions of this section include:

- We generate the rational Krylov subspace method for the computation of MEVPs due

to its flexibility on stiff systems and fast convergence rate. We apply the exponential related φ functions to improve the numerical accuracy of MEVPs.

- We exclude the Newton-Raphson iterations in the explicit matrix exponential based integration method. The convergence of the solution is checked by compensation iteration with an correction term evaluated by results from Krylov subspace.

- We devise the simulation framework which only solves the linear system once at each time step.

In modern VLSI designs, the power integrity becomes a critical issue to ensure the reliability and performance of designs. The challenges of power integrity analysis arise from the tighter noise margin with reducing power supply voltage, higher resistance on metal wires due to scaling, and strong coupling noise between the active devices.

The simulation of power integrity analysis encounters the problems from the increasing size of power delivery networks (PDN) as well as the accuracy of load models. Due to the increasing design complexity, the PDNs could be extremely huge and stiff, which makes the simulation a critical task. To simplify the system-level power integrity analysis, the on-chip macrocells are usually characterized as independent current sources with linear elements. However, the accuracy of power grid analysis is lost and the results could be far from the real cases. An efficient simulation framework is in high demand to handle the issues.

In Chapter 3.2 and Chapter 4, we have discussed in detail about the advantages of matrix exponential based integration methods for the transient simulation of linear PDNs. Adaptive stepping scheme can be used to speedup the simulation while maintaining the stability and accuracy. However, the performance of the framework is limited by the nonlinearity of systems. In Sec. 5.3, we propose a nonlinear macrocell model to capture the dynamic behavior of PDNs and we take advantage of the recent progress in the parallel-

in-time approach, such as Parareal (Parallel in Real time) [29] and MGRIT (Multigrid Reduction in Time) [14], and applied the idea to the PDN transient simulations. The main contributions are listed as follows

- We adopt a nonlinear voltage-dependent macrocell model in the PDN simulation framework to characterize the dynamic behaviors of whole systems.
- We apply the parallel-in-time method to parallelize the conventional sequential time stepping of the PDN transient simulation.
- We use the adaptive Newton-Raphson (NR) method to solve the nonlinear system efficiently in the iterations of step integrations.

5.2 Solving Nonlinear Systems with Numerical Integration Methods

5.2.1 Exponential Integrators in Nonlinear Circuit Simulation

Consider the nonlinear system after linearization in Eq. 2.2, we use k as the subscript notation for the variables at the kth time point t_k. With the initial variable x_k at t_k and step size h, we need to solve x_{k+1} at $t_{k+1} = t_k + h$ which satisfies Eq. 2.2. Solving such nonlinear systems is not trivial, instead we solve the linearized equation

$$\mathcal{C}_k \dot{x}_{k+1} + \mathcal{G}_k x_{k+1} = Bu(t_k + h) + F_k. \tag{5.1}$$

Since the nonlinear elements in $\mathcal{C}_k, \mathcal{G}_k$, and F_k are evaluated from the device models according to x_k, we can use the integration methods discussed in previous sections to solve the equations. MEVPs calculated with Krylov subspace provide a high order polynomial

approximation of the solution in Eq. 2.8 and the stability is preserved [4, 47, 64]. Notice that the term F_k is included in the input vectors but does not affect the whole formulation.

To improve the performance on stiff systems, we adopt the rational Krylov subspace [3, 13, 41, 53]. The exponential related φ functions are included to express the MEVPs for numerical concerns [4, 25, 49]. The derivations and algorithms can be found in Sec. 2.4 and Sec. 4.1.1.

5.2.2 Approximation Theory and Compensation Iteration for Convergence

The original Eq.2.2 enforces KCL and KVL laws at time t_k and t_{k+1}. In order to evaluate the undershoot or overshoot, a term Δx_{k+1} is used to express the difference between the approximated solution x_{k+1} from Eq. 5.1 and the real solution. With the extra correction term, the solution aims to satisfy

$$\mathcal{C}_{k+1}(\dot{x}_{k+1} + \Delta\dot{x}_{k+1}) + \mathcal{G}_{k+1}(x_{k+1} + \Delta x_{k+1}) = F_{k+1} + Bu(t_k + h), \tag{5.2}$$

which is equivalent to the following relation

$$\mathcal{C}_{k+1}\Delta\dot{x}_{k+1} + \mathcal{G}_{k+1}\Delta x_{k+1} = -\mathcal{C}_{k+1}\dot{x}_{k+1} - \mathcal{G}_{k+1}x_{k+1} + F_{k+1} + Bu(t_k + h). \tag{5.3}$$

which forms the DAEs of variable Δx_{k+1}. So Δx_{k+1} could be calculated similarly with MEVPs. The right hand side of the above equation is defined as the negative residual r_{k+1} of Eq. 2.2. If we treat the term as part of the ramp input [56],

$$\Delta u(t_k + h) \approx -\frac{1}{h} r_{k+1}, \tag{5.4}$$

then Δx_{k+1} will be added to original solution as a compensation term

$$\begin{aligned} x_{k+1} &= x_{k+1} + \Delta x_{k+1} \\ &\approx x_{k+1} + h^2 \varphi_2(h\mathcal{A})\mathcal{C}_k^{-1} \Delta u(t_k + h). \end{aligned} \tag{5.5}$$

The process is repeated until the solution converges. All the parameters with subscript $k+1$ in above derivation are evaluated by device models according to the updated x_{k+1}. Since x_k converges at t_k, the residual of Eq.5.1 should be negligible (below tolerance). We can find that r_{k+1} updates the solution due to change of nonlinear system

$$r_{k+1} \approx \Delta\mathcal{C}_{k+1}\dot{x}_{k+1} + \Delta\mathcal{G}_{k+1}x_{k+1} - \Delta F_{k+1}, \tag{5.6}$$

where $\Delta\mathcal{C}_{k+1} = \mathcal{C}_{k+1} - \mathcal{C}_k$, $\Delta\mathcal{G}_{k+1} = \mathcal{G}_{k+1} - \mathcal{G}_k$ and $\Delta F_{k+1} = F_{k+1} - F_k$. Therefore, the compensation term is the response to the change of the system's nonlinear elements from stage x_k to x_{k+1}. Algorithm 5 provides an iteration process showing how the correction term works to achieve convergence.

The LU decomposition is performed on $(\gamma\mathcal{G}_k + \mathcal{C}_k)$ for rational Krylov subspace. The MEVPs in solution are evaluated with appropriate φ functions. Lines 6-10 show the compensation iteration for circuit nonlinear elements. The residue term r_{k+1} is element-wisely compared to an error bound Err. Once the relation $r_{k+1} \leq Err$ is not satisfied, compensation term is computed and added to x_{k+1} until solution converges.

The framework also incorporates an adaptive step strategy. If the solution cannot converge within $Iter_{max}$ iterations, the time step h is shrunk and the solution has to be recalculated. IF the solution converges in a small number of iterations, the step size h will be increased for the next time step to accelerate the simulation process.

Algorithm 5: Integration Kernel for rational Krylov subspace using Compensation Iteration

Input: Circuit netlist, input sources, x_k at time t_k and expected time step h
Output: solution x_{k+1} at $t_k + h$

1 Load the device models and update $\mathcal{C}_k$, $\mathcal{G}_k$, F_k with x_k;
2 Perform LU decomposition on required matrices;
3 Apply the **Arnoldi** algorithm to compute the MEVPs and get the solution $x_{k+1}^{(0)}$;
4 Compute the stating residual $r_{k+1}^{(0)}$;
5 Set iteration number $i = 0$;
6 **while** $r_{k+1}^{(i)} > Err$ *not converged and* $i < Iters_{max}$ **do**
7 Compute compensation term in Eq. 5.3; Update $x_{k+1}^{(i+1)} = x_{k+1}^{(i)} + \Delta x_{k+1}$;
8 Load the device models and compute r_{k+1}^{i+1} with $x_{k+1}^{(i+1)}$;
9 Set $i = i + 1$;
10 **end**
11 **if** $r_{k+1}^{(i)} > Err$ **then**
12 $i = 0$; $h = \mu h$; `// Computed solution` x_{k+1} `is rejected. Shrink` h `by` $\mu < 1$ `and redo from line 3.`
13 **end**
14 **else**
15 $x(t_k + h) = x_{k+1}^{i}$; $t = t + h$; $k = k + 1$;
`// Solution` x_{k+1} `is converged.`
16 **if** $i \leq Iters_{min}$ **then**
17 $h = \alpha h$; `//` i `is small,` h `is increased by` $\alpha > 1$ `to accelerate the process.`
18 **end**
19 **end**

5.2.3 Experimental Results

In this section, we apply our proposed explicit matrix exponential based integration with compensate iterations to the transient simulation of analog circuits. Table 5.1 provides the specification of the nonlinear test cases from industry. We define $\gamma = h/2$ as default value and restrict the maximum allowed step within $1ns$. In order to verify the numerical difference among MEVPs with φ functions, all test cases are stiff designs with nonsingular $\mathcal{C}$. No regularization process is needed for the ODEs. Size of the test cases varies from 43 to 40k, represented by #Node. #Dev is the number of MOSFETs in each circuit. The nonzero

elements of system matrices are included, we can find that D4 - D6 have relatively denser matrices. T is the total simulation time. Same tolerance is set in the simulation framework to all experiments. Convergence of nonlinear system is achieved using the compensation iteration with a correction term. The MEVPs are evaluated with φ_0, φ_1 and φ_2 functions separately. Implementations can be found in Sec. 4.1.1.

Table 5.1: Specifications of Analog designs

Index	Design	#Node	#Dev	nnz(C)	nnz(G)	T (s)
D1	voter25	43	74	345	345	1×10^{-6}
D2	counter	93	220	0.7k	0.7k	1×10^{-7}
D3	fadd32	161	288	1.1k	1.1k	1×10^{-7}
D4	add20	521	958	7.2k	3.6k	1.6×10^{-7}
D5	memplus	2.8k	7.4k	35k	26k	4.75×10^{-7}
D6	ram2k	4.8k	13.8k	47.6k	47.6k	6×10^{-7}
D7	Inv. chain	11k	24	63k	34k	1×10^{-9}

The simulation results are listed in Table 5.2. DC represents the DC analysis time of each test case. Average dimension of rational Krylov subspace is denoted as m_a, which includes the computations of solution and residual term in compensation iteration. Total time steps and runtime are displayed as well. $Iter_{avg}$ is the average number of iterations, which reflects the convergence rate of circuits. Designs with more complex matrices tend to have larger $Iter_{avg}$, like D4 - D6. Relatively high m for Krylov subspace is observed for those cases. For all the test cases, φ_0 method costs the shortest running time with smallest m_a. To achieve same accuracy, smaller m of Krylov subspace and less computation is required with φ_0. The reason is that the step sizes in the simulation lies in the spectrum of $\mathcal{G}^{-1}\mathcal{C}$ or larger than the spectrum.

In Fig. 5.1, the waveform of D4 is extracted to compare with the traditional BE method with NR iteration (BENR). Smaller time step ($0.1ps$) is applied to BENR to calculate the reference solution. Solution computed with our proposed algorithm well fits the reference.

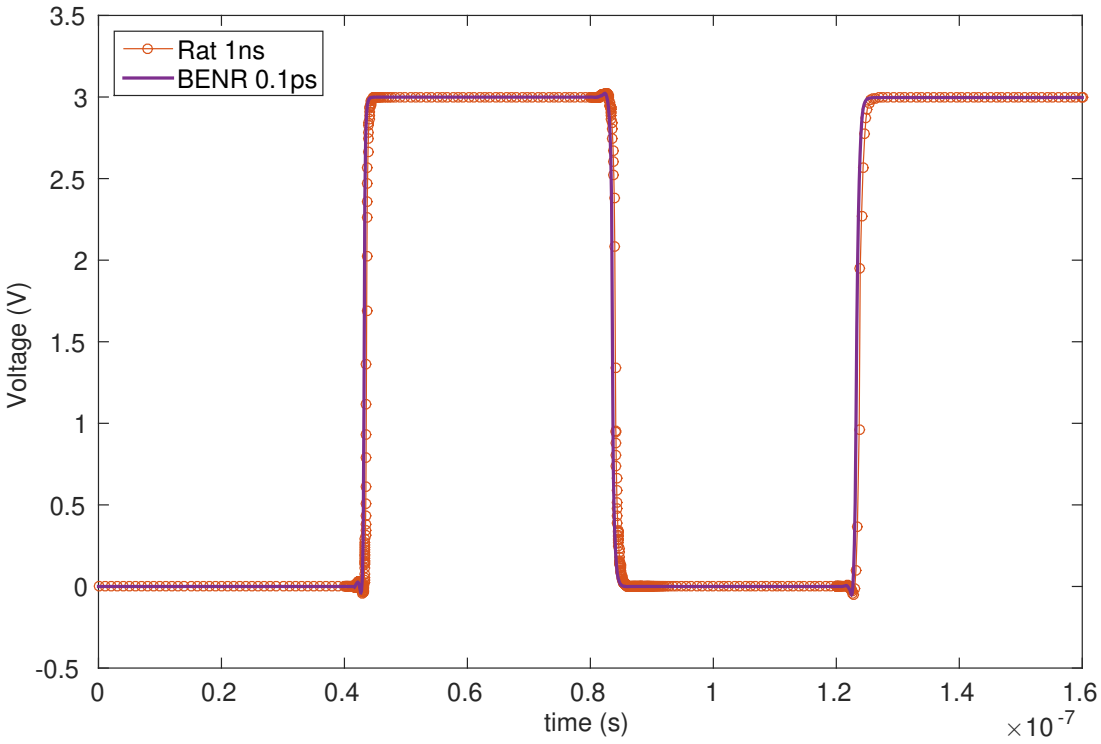

Figure 5.1: Transient simulation results of a nonlinear circuit by matrix exponential integration with compensate iteration and BENR method.

We implement the algorithms for circuit transient simulation in MATLAB 2014a and use UMFPACK package for matrix factorization. The experiments are performed on a Linux server with Intel(R) Xeon(R) CPU E5-2640 v3 2.60GHz and 125 GB memory. Device evaluation and matrix stamping are done in C/C++ with BSIM3 model for MOSFET. The interactions are through MATLAB Executable (MEX) external interface with GCC 4.4.7.

5.3 Parallel-in-time Methods for PDN Transient Simulation with Nonlinear Load Models

To improve the accuracy of PDN transient simulations, we propose a nonlinear power load model to characterize the variations of the dynamic cell behaviors. The transient

Table 5.2: Simulation Performance of Rational Krylov Subspace method with Exponential Integrators

Design	DC(s)	φ_0 method				φ_1 method				φ_2 method			
		m_a	Step	Tran(s)	$Iter_{avg}$	m_a	Step	Tran(s)	$Iter_{avg}$	m_a	Step	Tran(s)	$Iter_{avg}$
D1	0.01	24.3	3896	45.3	1.93	34.1	3896	63.6	1.93	37.7	3891	72.3	1.92
D2	0.28	23.7	344	4.4	0.78	31.0	344	6.9	0.79	32.0	344	6.9	0.79
D3	0.02	25.0	844	14.3	2.24	33.5	844	19.9	2.24	37.9	844	21.2	2.24
D4	0.33	31.2	734	36.1	3.22	42.3	734	42.8	3.22	55.5	733	53.6	3.17
D5	1.39	24.9	1853	496.7	1.89	33.1	1859	568.7	1.86	46.4	1861	703.6	1.91
D6	1.75	29.1	3265	1633	1.88	40.3	3191	1975	1.94	47.1	3168	2071	1.97
D7	0.13	13.9	180	60.2	1.12	20.4	180	98.3	1.12	18.2	180	86.3	1.12

simulation is more challenging.

Most of the previous parallel simulation works relies on the distributed matrix solvers, e.g. Xyce [54]. The integration in transient simulation is still operated in series. The solution at time $t + h$ with step size h has to wait until the solution at previous time t is ready. The parallel-in-time method approximates the solution on coarse time grids and refines the solution with fine grid approximations which are performed in parallel. Compared to the conventional sequential stepping scheme, the proposed parallel method obtains speed up on wall time but uses more iterations globally.

5.3.1 Nonlinear Load Models in PDNs

In general, linear current source model [5] is used in power integrity analysis due to the complexity of millions to billions devices. However, the variation of instance switching current caused by dynamic IR drop in PDN can not be ignored. We propose a nonlinear macrocell load model to include the effects of Dynamic Voltage Drop (DvD) in the PDNs [19, 26, 62]. A voltage dependent current source $I_{load}(t, v_{sup})$ with series RC, $R_{load}(v_{sup})$ and $C_{load}(v_{sup})$, are used to model the current fluctuation caused by DvD at the power supply node (i.e., v_{sup}), as shown in Fig. 5.2. Our nonlinear load model provides the fixed pivot points information, which enable us to determine the simulation time points in advance.

The nonlinear load models are generated at different supply voltages (i.e., V_j). During the transient simulation, the values of elements can be interpreted based on v_{sup} at

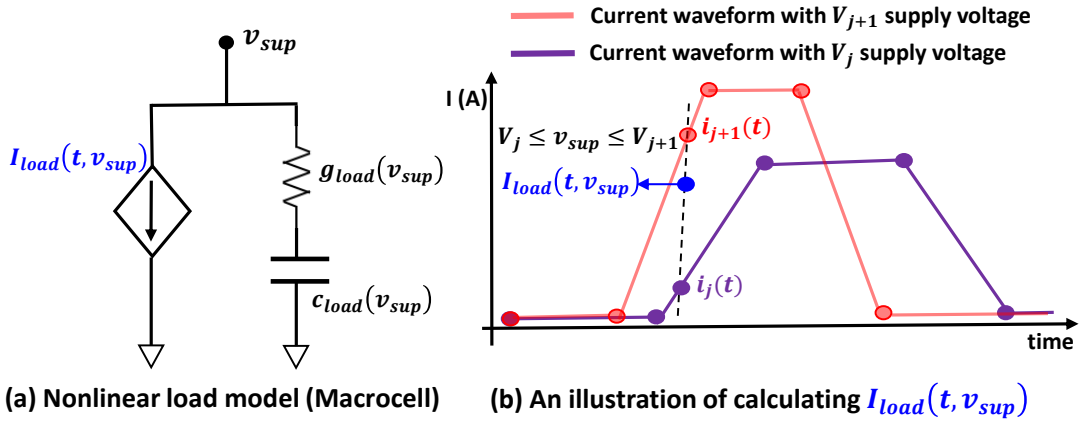

Figure 5.2: An illustration of the nonlinear load model in PDN. The dynamic behaviors of macrocells are characterized with voltage dependent current source and RC in series.

t as

$$\begin{aligned} I_{load}(t, v_{sup}) &= i_j(t) + \frac{(i_{j+1}(t) - i_j(t))}{(V_{j+1} - V_j)}(v_{sup} - V_j) \\ g_{load}(v_{sup}) &= g_j + \frac{(g_{j+1} - g_j)}{(V_{j+1} - V_j)}(v_{sup} - V_j) \\ c_{load}(v_{sup}) &= c_j + \frac{(c_{j+1} - c_j)}{(V_{j+1} - V_j)}(v_{sup} - V_j) \end{aligned} \tag{5.7}$$

where v_{sup} lies between two supply voltages $[V_j, V_{j+1}]$ and the coefficients i, g, c represent the element values at each supply voltage in the macrocell model.

We revisit the DAE formulation of PDNs with the nonlinear macrocell models.

$$\mathcal{C}(x)\dot{x}(t) + \mathcal{G}(x)x(t) = Bu(x, t), \tag{5.8}$$

where the elements in system matrices $\mathcal{C}, \mathcal{G}$, and the input $u(t)$ are functions of variable x.

5.3.2 MGRIT method with Linear Step integrators

We propose a parallel-in-time method for nonlinear PDN transient simulations with the MGRIT method [14] and adaptive Newton-Raphson techniques, named as *MGRIT-AdapNR*. Firstly, we discuss the application of the MGRIT method to circuit simulation in this section. In the next Sec. 5.3.3, we introduce the adaptive NR method to solve the nonlinear equations.

Parareal was first presented as a numerical method to solve evolution problems [29] and extended to PDEs with many follow up works [14, 16], which enrich the field. Consider the DAEs in Eq. 5.8, BE integration starts from

$$x(t+h) = x(t) + h\dot{x}(t+h), \tag{5.9}$$

which gives

$$\big(\frac{\mathcal{C}(x)}{h} + \mathcal{G}(x)\big)x(t+h) = \frac{\mathcal{C}(x)}{h}x(t) + Bu(x, t+h), \tag{5.10}$$

where we can define the operator $M = \big(\frac{\mathcal{C}(x)}{h} + \mathcal{G}(x)\big)^{-1}$ on the rhs of equation at t. The DAEs can be solved with linear step integration method with NR iterations.

Fig. 5.3 demonstrate (1) the general sequential integration method and (2) the two level MGRIT method. For MGRIT method, we define the fine and coarse time grids. We assume the fine time grids have uniform step size h. Each time interval in the coarse time grid equals multiple steps of the fine time grids with $H = Mh$. We define two integrators on the two levels. Let $M_L(T_{n+1}, T_n, x_n)$ denote the long step integration on the coarse time grid from T_n to T_{n+1}, where $T_n = t_{Mn}$. Let $M_S(T_{n+1}, T_n, x_n)$ denote the short step integration on the fine grid which takes M steps from T_n to T_{n+1}. The MGRIT method

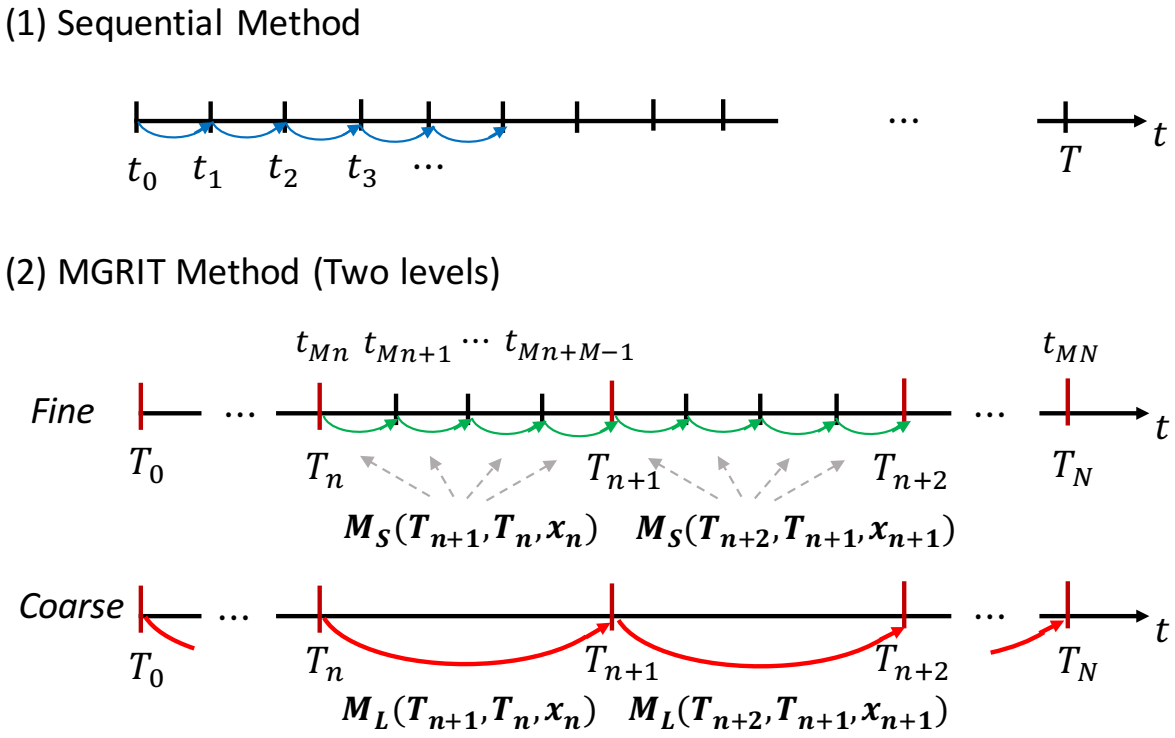

Figure 5.3: In PDN transient simulation, step integrators are applied to (1) general sequential method and (2) MGRIT method with two levels.

performs k iterations and approximates the next approximation with the formulation:

$$x_{n+1}^{(k+1)} = M_L(T_{n+1}, T_n, x_n^{(k+1)}) + M_S(T_{n+1}, T_n, x_n^{(k)}) - M_L(T_{n+1}, T_n, x_n^{(k)}). \tag{5.11}$$

Note that in Eq. 5.11, the first long step integration includes the solution x_n^{k+1}. Therefore, it has to be performed sequentially in order to get the initial condition in current iteration. The second and the third term only depends on results from the previous iteration, the integrations in $M_S(T_{n+1}, T_n, x_n^{(k)})$ and $M_L(T_{n+1}, T_n, x_n^{(k)})$ between any time interval can be operated in parallel.

In [18], the Parareal algorithm is illustrated as a multigrid in time method when reduced to two levels. The long and short step integrators are defined on coarse and fine time grids, respectively. We apply the ideas from works [14,17] to generalize the parallelism

to multiple temporal levels.

To illustrate the MGRIT as a time-multigrid method, we consider a nonlinear system on the fine grid

$$A^h(x^h) = b. \tag{5.12}$$

To clarify the notations on two grids, we use x_m^h to denote the solution on fine grid with index m, where the fine time grids are $[t_0, t_1, \ldots, t_{MN}]$ with step size h. While the solution on coarse grid is denoted with X_n^H on $[T_0, T_1, \ldots, T_N]$, where $T_n = t_{Mn}$ and $X_n^H \approx x_{Mn}^h$. Similarly, the coarse grid problem is characterized by

$$A^H(X^H) = B. \tag{5.13}$$

The two grid full approximation storage(FAS) multigrid method is written as Algorithm 6. The algorithm solves the time series of approximations x_m^h for $m = 1, 2, \ldots,$ starting

Algorithm 6: Two Grid FAS Algorithm for MGRIT

1 **while** *norm of residual is not small enough* **do**
2 Smooth/Relax on the fine grid: $A^h(x_m^h) = b_m$ with $m = 1, 2, \ldots, MN$.
3 Compute the residual: $r_{Mn}^h = b_{Mn} - A^h(x_{Mn}^h)$.
4 Restrict the residual to coarse grid: $f_n^H = I_h^H r_{Mn}^h$.
5 Solve the coarse grid problem: $A^H(X_n^H) = A^H(I_h^H x_{Mn}^h) + f_n^H$.
6 Compute the correction on coarse grid: $e_n^H = X_n^H - I_h^H x_{Mn}^h$.
7 Update the fine grid solution: $x_{Mn}^h = x_{Mn}^h + I_H^h e_n^H$.
8 **end**

from a given initial x_0. The residual and the approximation on fine grid are transferred to the coarse grid with the restriction operator I_h^H. Then, a correction term is calculated

based on the coarse grid problems, which is used to fix the solution on coarse grid with the prolongation operator I_H^h.

The operators A^h and A^H can be expressed with single step integrators used on fine an coarse grids, respectively. We define the single step integration method for the fine grid approximation,

$$x_m^h = \phi_m(x_{m-1}^h) + b_m, m = 1, 2, \ldots, MN, \tag{5.14}$$

and for the coarse grid approximation,

$$X_n^H = \Phi_n(X_{n-1}^H) + B_n, n = 1, 2, \ldots, N. \tag{5.15}$$

The short step integrator includes M steps of Eq. 5.14 from x_{nM}^h to $x_{(n+1)M}^h$,

$$M_S(T_{n+1}, T_n, X_n^H) := x_{(n+1)M}^h \text{ where } X_n^H = x_{nM}^h. \tag{5.16}$$

The long step integrator includes one step of Eq. 5.15,

$$M_L(T_{n+1}, T_n, X_n^H) := \Phi_{n+1}(X_n^H) + B_{n+1}. \tag{5.17}$$

For a linear system, the Eq. 5.12 can be written as a matrix format,

$$A^h = \begin{pmatrix} I & & & \\ -\phi_1 & I & & \\ & \ddots & \ddots & \\ & & -\phi_{MN} & I \end{pmatrix}, x^h = \begin{pmatrix} x_0 \\ x_1 \\ \cdots \\ x_{MN} \end{pmatrix}, b = \begin{pmatrix} b_0 \\ b_1 \\ \cdots \\ b_{MN} \end{pmatrix}, \tag{5.18}$$

which can be used for the coarse grid problem as well.

The interpretation of Parareal/MGRIT as a time multigrid method is well illustrated in previous work, detailed proof can be found in [18]. The iteration in MGRIT is consistent with the fine grid problem and the algorithm follows the linear convergence of multigrid methods [10, 17, 18].

5.3.3 Nonlinear Systems and Adaptive Newton-Raphson Iterations

For a nonlinear system, the implicit formulation Eq. 5.8 requires NR iterations to achieve a converged solution. We define the residual of the system at t as

$$r(x) \approx Bu(x,t) - \mathcal{C}(x)\dot{x}(t) - \mathcal{G}(x)x(t). \tag{5.19}$$

Based on the Taylor expansion around the current approximation $x^{(k)}$, the next approximation $x^{(k+1)}$ satisfies

$$0 = r(x^{(k+1)}) \approx r(x^{(k)}) + J(x^{(k)})(x^{(k+1)} - x^{(k)}), \tag{5.20}$$

where $J(x)$ is the Jacobian matrix with $J_{ij}(x) = \frac{\partial r_i}{\partial x_j}$. In practical circuit simulation, the $J(x)$ is given by the nonlinear elements and choice of multi-step method. The NR iterations follow the relation

$$x^{(k+1)} = x^{(k)} - J(x^{(k)})^{-1} r(x^{(k)}). \tag{5.21}$$

The corresponding Jacobian is updated at each iteration according to $x^{(k)}$. Either the residual $r(x^{(k+1)})$ is below given tolerance or the change of solution from $x^{(k)}$ to $x^{(k+1)}$ is small enough the iterations are terminated.

Unlike the traditional method where NR iterations are used at each step, adaptive

NR (adap. NR) method skips the NR iterations if the change of x at $t+h$ satisfies

$$\|\Delta x^{(0)}\|_\infty \leq \Delta_{th}, \tag{5.22}$$

where $\Delta x^0 = x^{(0)}(t+h) - x(t)$ and Δ_{th} is the given threshold. Considering the nonlinear macrocell model is less sensitive to its voltage than transistors, we can set larger Δ_{th} to improve the performance.

5.3.4 Experimental Results

The *MGRIT-AdapNR* is implemented via the open source software library Xbraid [1] in C++. All experiments are performed on a 1.8GHz Intel Xeon 24-CPUs server.

Table 5.3 shows the statistics of PDNs with size ranges from thousands to millions, where the design "genckt30" is created based on the specifications in [38] and used for optimum parameter exploration. For ibmpg1t-nl, ibmpg2t-nl, and ibmpg3t-nl, we extend the original power loads to nonlinear load models with the guidance from industry and use the original PDNs of ibmpg1t, ibmpg2t, and ibmpg3t [38]. The nonlinear load models are updated using Eq. 5.7 in the transient simulations. We compare *MGRIT-AdapNR* with Sequential solver (*Seq*) using NR iterations at each time step. The maximum absolute error e_{max} and average absolute error e_{avg} are calculated from the probing nodes of each design and reported in the following experiments. The runtime represents the wall time.

Table 5.3: Design specifications of PDNs

PDN	#R	#C	#L	#Loads	#Size	#Probing Nodes
genckt30	2.6K	1.4K	0	720	1.6K	90
ibmpg1t-nl	54K	11K	277	11K	40K	24
ibmpg2t-nl	245K	37K	330	37K	165K	20
ibmpg3t-nl	1.6M	201K	955	201K	1M	20

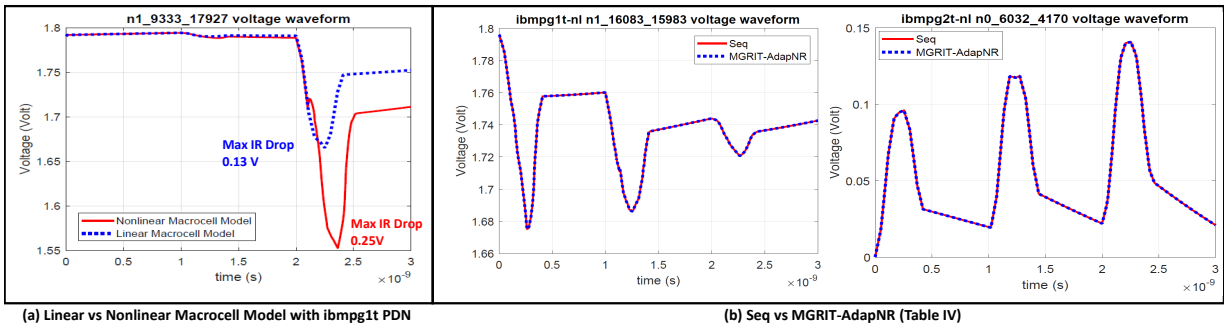

Figure 5.4: (a) Linear vs Nonlinear Macrocell Model; (b) Nodal waveforms of *Seq* and *MGRIT-AdapNR* (Table 5.6).

Study I: Linear vs Nonlinear Load Model

Fig. 5.4(a) shows the simulation results of a nodal waveform from ibmpg1t with linear load models and ibmpg1t-nl with nonlinear load models. The simulation time is 3ns with 900 time steps. The maximum IR drop with nonlinear load models is 92% larger, which is underestimated by the linear models. The nonlinear load model is essential for power integrity analysis.

Study II: Optimum Parameter Exploration

We perform multiple experiments on #Cores, Coarsening Factor (CF), and Maximum Level (ML) of *MGRIT-AdapNR* to find the optimum settings in terms of runtime and accuracy.

#Cores: Table. 5.4 shows the *MGRIT-AdapNR* runtime with 4, 8, 16, and 24 cores. Compared to the *Seq*, *MGRIT-AdapNR* with 24 cores achieves 2× speedup and the max error is $3mV$. Besides, the difference in max and avg errors of different #Cores is less than 1%. *MGRIT-AdapNR* is robust with various #Cores.

Coarsening Factor (CF) and Maximum Level (ML): We use "genckt30" with 410K time steps to explore the optimum CF and ML to fully leverage the parallel-in-time advantage. Table 5.5 shows the results of *Seq* and *MGRIT-AdapNR* with various combinations

Table 5.4: Experimental results of different #Cores using ibmpg1t-nl with 3ns simulation time and 900 time steps.

	#Cores	e_{max} (mV)	e_{avg} (mV)	Runtime (s)	Speedup (X)
Seq	1	-	-	4790.61	1
MGRIT-AdapNR	4	3.00	3.82E-3	6882.63	0.70
	8	3.00	3.83E-3	4250.47	1.13
	16	3.00	3.84E-3	3092.15	1.55
	24	3.00	3.84E-3	2493.53	1.92

of CF and ML. We select CF=2, 6, and 10. Then, we increase the ML from 2 to 10 with increment 2 until the time grids cannot be coarsened any more. From the results, CF=10 and ML=4 achieves the best performance. The max error is less than 1mV.

Table 5.5: Experimental results of *Seq* and *MGRIT-AdapNR* (24 cores), with multiple combinations of CF and ML using genckt30 test case. Simulation time=6ns. #time steps=410K. Time Grid Ratio=(#Finest Time Grids)/(#Coarsest Time Grids).

Method	CF	ML	Time Gird Ratio	e_{max} (mV)	e_{avg} (mV)	Runtime (s)	Speedup (X)
Seq	-	-	1	-	-	1289.07	1
MGRIT-AdapNR	2	2	2	0.01	4.11E-4	1967.61	0.66
		4	8	0.06	5.86E-3	1011.81	1.27
		6	32	0.16	1.02E-2	793.43	1.62
		8	128	0.22	1.02E-2	730.05	1.77
		10	512	0.22	1.04E-2	710.14	1.82
	6	2	6	0.04	5.75E-3	938.4	1.37
		4	216	0.36	5.84E-3	445.83	2.89
		6	1296	1.10	2.91E-2	426.2	3.02
	10	2	10	0.07	7.70E-3	730.18	1.77
		4	1000	0.14	1.10E-2	390.74	3.30
		6	100000	5.60	2.86E-1	387.27	3.33

Main Results

Table 5.6 shows our main results on PDNs in Table 5.3. The simulation time of ibmpg1t-nl, ibmpg2t-nl, and ibmpg3t-nl are 3ns, 3ns, and 2ns with 900, 960, and 630 time steps, respectively. The *MGRIT-AdapNR* multigrid cycles of all three cases are 3. The

MGRIT-AdapNR multigrid cycles and Adap. NR reduce the #NewtonIters up to 30%. Compared to *Seq*, *MGRIT-AdapNR* achieves more than 2× speedup with less than 5mV max error. The *MGRIT-AdapNR* successfully captures the transient waveform of nonlinear PDNs, as shown in Fig. 5.4(b).

Table 5.6: Experimental results of *Seq* and *MGRIT-AdapNR* (#Core=24, CF=10 and ML=4).

	e_{max} (mV)	e_{avg} (mV)	#NewtonIters		Runtime (s)		Speedup (X)
			Seq	Proposed	Seq	Proposed	
ibmpg1t-nl	3.00	3.84E-3	1982	1521	4790.61	2493.53	1.92
ibmpg2t-nl	3.40	8.24E-2	2304	1662	17882.07	7947.37	2.25
ibmpg3t-nl	2.54	3.12E-2	1824	1256	102683.35	43430.18	2.36

5.4 Summary

In Sec. 5.2, we propose an efficient algorithmic framework for the nonlinear circuit time domain simulation using exponential integrators. The MEVPs are computed by rational Krylov subspace. In order to reduce the number of LU decomposition operations, we exclude the Newton-Raphson iterations with the explicit integration method. A residual based compensation iteration is devised to maintain the convergence. The φ functions are applied to the MEVPs of nonlinear circuits. Sec. 5.2.3 shows that by choosing appropriate φ functions for the computation of MEVPs we can achieve low computation cost while ensuring accuracy.

In Sec. 5.3, we develop the *MGRIT-AdapNR* for the transient analysis of PDNs with nonlinear load models, where the time integration is parallelized. Compared to the *Seq*, *MGRIT-AdapNR* achieves 3× speedup on long simulation time (410K time steps) and 2× speedup on the PDNs from 40K to 1M size. Without the limitation of maximum #Cores on our server, we expect that *MGRIT-AdapNR* can achieve more speedups. The future research directions include (i) exploring the performance improvement of *MGRIT-AdapNR*

with more cores and (ii) improving the convergence rate using advance integrators such as Matrix Exponential [55].

Chapter 5, in part, is a reprint of the material in the work: X. Wang, H. Zhuang, and C. K. Cheng, "Exploring the exponential integrators with krylov subspace algorithms for nonlinear circuit simulation," in *Proceedings of the 36th International Conference on Computer-Aided Design*, pages 163–168. IEEE Press, 2017. The author is the primary author and investigator of this work. The chapter also contains the submission for publication of the material in the work: C.-K. Cheng, C.-T. Ho, C. Jiao, X. Wang, Z. Zeng, and X. Zhan, "A parallel-in-time circuit simulator for power delivery networks with nonlinear load models," submitted to *the 29th Conference on Electrical Performance of Electronic Packaging and Systems*, 2020. The author is one of the primary authors and investigators of this work.

Chapter 6

Conclusions

This chapter summarizes the contributions of this thesis and presents a future scope for the advanced transient simulation algorithms of VLSI circuits.

Chapter 2 introduces the formulation of circuit transient simulations and the linear multi-step integration methods. The matrix exponential based integration method is proposed evaluated by standard, invert, and rational Krylov subspace. The numerical performance of the matrix exponential integration is compared with the conventional integration methods, which demonstrate its advantages for circuit transient simulation.

Chapter 3 focuses on the stability of matrix exponential based integration algorithms when used to solve the DAEs in PDN transient simulations. We prompt a modified Arnoldi algorithm with structured orthogonalization to produce stable results of rational Krylov subspace. The singular capacitance matrix is used to induce the orthogonality as well as implicitly regularize the DAEs by excluding the algebraic equations. Significantly improved stability and accuracy have been observed on general RC and RLC networks.

Chapter 4 presents the total simulation framework with matrix exponential based integration methods. The exponential related φ functions are investigated to improve the computation of MEVPs. In the construction of rational Krylov subspace, we also explore

the optimal ratio to confine the spectrum of original system. With the stable Aronoldi algorithm and the devised techniques, we apply adaptive stepping strategy to the transient simulation of system-level PDNs. Simulation results on general PDN cases are consistent to our experiments.

Chapter 5 presents the simulation techniques for nonlinear system with different scopes. First, we apply the explicit matrix exponential based integration to the analog designs with strong nonlinearity. We remove the Newton-Raphson iterations to avoid extra matrix factorization in the simulation, instead, a residual based compensate iteration is introduced for convergence. Second, we explore the parallel-in-time methods to the transient simulation of PDNs with nonlinear load models, which breaks the conventional sequential time stepping scheme and provides a promising direction for speedup of nonlinear systems.

There are many open problems to be explored in the future.

- The general applications of matrix exponential to simulation of circuits in mixed signal analysis, logic synthesis, power and timing analysis, thermal analysis, etc.
- Speedup of the transient simulation with nonlinear system remains to be an interesting topic. The performance of parallel-in-time method on circuit simulation is not fully explored.
- Explore more advanced technologies from other areas to the circuit simulation.

Bibliography

[1] Xbraid: Parallel multigrid in time. `http://llnl.gov/casc/xbraid`.

[2] A. H. Al-Mohy and N. J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM journal on scientific computing*, 33(2):488–511, 2011.

[3] M. A. Botchev. A short guide to exponential krylov subspace time integration for maxwell's equations. Dept. of Applied Mathematics, Univ. of Twente, 2012.

[4] M. Caliari and A. Ostermann. Implementation of exponential rosenbrock-type integrators. *Applied Numerical Mathematics*, 59(3-4):568–581, 2009.

[5] H. H. Chen and D. D. Ling. Power supply noise analysis methodology for deep-submicron vlsi chip design. In *Proceedings of the 34th annual Design Automation Conference*, pages 638–643, 1997.

[6] P. Chen, C. K. Cheng, D. Park, and X. Wang. Transient circuit simulation for differential algebraic systems using matrix exponential. In *Proceedings of the International Conference on Computer-Aided Design*, page 99. ACM, 2018.

[7] Q. Chen, S.-H. Weng, and C. K. Cheng. A practical regularization technique for modified nodal analysis in large-scale time-domain circuit simulation. *IEEE TCAD*, 31(7):1031–1040, 2012.

[8] L. O. Chua and P.-M. Lin. *Computer Aided Analysis of Electric Circuits: Algorithms and Computational Techniques*. Prentice-Hall, 1975.

[9] W. Cody, G. Meinardus, and R. Varga. Chebyshev rational approximations to e- x in [0,+) and applications to heat-conduction problems. *Journal of Approximation Theory*, 2(1):50–65, 1969.

[10] V. Dobrev, T. Kolev, N. A. Petersson, and J. B. Schroder. Two-level convergence theory for multigrid reduction in time (mgrit). *SIAM Journal on Scientific Computing*, 39(5):S501–S527, 2017.

[11] W. Dong and P. Li. Parallelizable stable explicit numerical integration for efficient circuit simulation. In *Proceedings of IEEE/ACM Design Automation Conference*, 2009.

[12] W. Dong, P. Li, and X. Ye. Wavepipe: Parallel transient simulation of analog and digital circuits on multi-core shared-memory machines. In *Proc. IEEE/ACM Design Autom. Conf.*, pages 238–243, 2008.

[13] T. Ericsson and A. Ruhe. The spectral transformation lanczos method for the numerical solution of large sparse generalized symmetric eigenvalue problems. *Mathematics of Computation*, 35(152):1251–1268, 1980.

[14] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder. Parallel time integration with multigrid. *SIAM Journal on Scientific Computing*, 36(6):C635–C661, 2014.

[15] R. W. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123(1-2):395–421, 2000.

[16] S. Friedhoff and B. S. Southworth. On "optimal" h-independent convergence of parareal and mgrit using runge-kutta time integration. *arXiv preprint arXiv:1906.06672*, 2019.

[17] M. J. Gander and E. Hairer. Nonlinear convergence analysis for the parareal algorithm. In *Domain decomposition methods in science and engineering XVII*, pages 45–56. Springer, 2008.

[18] M. J. Gander and S. Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing*, 29(2):556–578, 2007.

[19] H. Harizi, R. HauBler, M. Olbrich, and E. Barke. Efficient modeling techniques for dynamic voltage drop analysis. In *2007 44th ACM/IEEE Design Automation Conference*, pages 706–711. IEEE, 2007.

[20] N. J. Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

[21] C. Ho, A. Ruehli, and P. Brennan. The modified nodal approach to network analysis. *IEEE TCAS*, 22(6):504–509, 1975.

[22] M. Hochbruck and C. Lubich. On krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.

[23] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM Journal on Scientific Computing*, 19(5):1552–1574, 1998.

[24] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[25] M. Hochbruck, A. Ostermann, and J. Schweitzer. Exponential rosenbrock-type methods. *SIAM Journal on Numerical Analysis*, 47(1):786–803, 2009.

[26] P.-Y. Hsu, C.-H. Yao, Y. Wang, and C.-K. Cheng. Adaptive sensitivity analysis with nonlinear power load modeling. In *2018 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, pages 1–6. IEEE, 2018.

[27] A. Ilchmann and T. Reis. *Surveys in differential-algebraic equations II*. Springer, 2014.

[28] P. Li. Parallel circuit simulation: A historical perspective and recent developments. *Foundations and Trends in Electronic Design Automation*, 5(4):211–318, 2012.

[29] J.-L. Lions, Y. Maday, and G. Turinici. A parareal in time discretization of pde's. In *C.R. Acad. Sci. Paris Ser. I Math*, pages 661–668, 2001.

[30] K. Meerbergen and A. Spence. Implicitly restarted arnoldi with purification for the shift-invert transformation. *Mathematics of Computation of the American Mathematical Society*, 66(218):667–689, 1997.

[31] Q. Mei, W. Schoenmaker, S.-H. Weng, H. Zhuang, C. K. Cheng, and Q. Chen. An efficient transient electro-thermal simulation framework for power integrated circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(5):832–843, 2016.

[32] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM review*, 20(4):801–836, 1978.

[33] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review*, 45(1):3–49, 2003.

[34] L. Nagel. *SPICE2: A computer program to simulate semiconductor circuits*. Ph.D. dissertation, 1975.

[35] L. Nagel and R. Rohrer. Computer analysis of nonlinear circuits, excluding radiation (CANCER). *IEEE Journal of Solid-State Circuits*, 6(4):166–182, 1971.

[36] L. W. Nagel and D. O. Pederson. *SPICE: Simulation program with integrated circuit emphasis*. Electronics Research Laboratory, College of Engineering, University of California, 1973.

[37] F. N. Najm. *Circuit simulation*. Wiley, 2010.

[38] S. R. Nassif. Power grid analysis benchmarks. In *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, pages 376–381. IEEE, 2008.

[39] K. Nichols, T. Kazmierski, M. Zwolinski, and A. Brown. Overview of spice-like circuit simulation algorithms. *IEE Proceedings-Circuits, Devices and Systems*, 141(4):242–250, 1994.

[40] J. Nissen and W. M. Wright. a krylov subspace algorithm for evaluating the phi function appearing in exponential integrators. *ACM Transactions on Mathematical Software*, 38(3):1–21, 2012.

[41] B. Nour-Omid, B. N. Parlett, T. Ericsson, and P. S. Jensen. How to implement the spectral transformation. *Mathematics of Computation*, 48(178):663–673, 1987.

[42] A. Odabasioglu, M. Celik, and L. T. Pileggi. Prima: Passive reduced-order interconnect macromodeling algorithm. In *Proceedings of the 1997 IEEE/ACM international conference on Computer-aided design*, pages 58–65. IEEE Computer Society, 1997.

[43] L. Orecchia, S. Sachdeva, and N. K. Vishnoi. Approximating the exponential, the lanczos method and an o (m)-time spectral algorithm for balanced separator. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1141–1160, 2012.

[44] L. T. Pillage, R. A. Rohrer, and C. Visweswariah. *Electronic circuit and system simulation methods*. McGraw-Hill New York, 1995.

[45] J. Rommes and N. Martins. Exploiting structure in large-scale electrical circuit and power system problems. *Linear Algebra and its Applications*, 431(3-4):318–333, 2009.

[46] A. Ruhe. Rational krylov sequence methods for eigenvalue computation. *Linear Algebra and its Applications*, 58:391–405, 1984.

[47] Y. Saad. Analysis of some krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 29(1):209–228, 1992.

[48] Y. Shi and L. He. Modeling and design for beyond-the-die power integrity. In *Proceedings of the International Conference on Computer-Aided Design*, pages 411–416. IEEE Press, 2010.

[49] R. B. Sidje. Expokit: A software package for computing matrix exponentials. *ACM Transactions on Mathematical Software*, 24(1):130–156, 1998.

[50] L. M. 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(3-4):377–389, 1999.

[51] B. Simeon, C. Führer, and P. Rentrop. The drazin inverse in multibody system dynamics. *Numerische Mathematik*, 64(1):521–539, 1993.

[52] M. Takamatsu and S. Iwata. Index characterization of differential–algebraic equations in hybrid analysis for circuit simulation. *International Journal of Circuit Theory and Applications*, 38(4):419–440, 2010.

[53] J. van den Eshof and M. Hochbruck. Preconditioning Lanczos approximations to the matrix exponential. *SIAM J. Sci. Comput.*, 27(4):1438–1457, 2006.

[54] J. Verley, E. R. Keiter, and H. K. Thornquist. Xyce: Open source simulation for large-scale circuits. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2018.

[55] X. Wang, P. Chen, and C.-K. Cheng. Stability and convergency exploration of matrix exponential integration on power delivery network transient simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[56] X. Wang, H. Zhuang, and C. K. Cheng. Exploring the exponential integrators with krylov subspace algorithms for nonlinear circuit simulation. In *Proceedings of the 36th International Conference on Computer-Aided Design*, pages 163–168. IEEE Press, 2017.

[57] G. Wanner. Dahlquist's classical papers on stability theory. *BIT Numerical Mathematics*, 46(3):671–683, 2006.

[58] S.-H. Weng, Q. Chen, and C. K. Cheng. Time-domain analysis of large-scale circuits by matrix exponential method with adaptive control. *IEEE TCAD*, 31(8):1180–1193, 2012.

[59] J. H. Wilkinson. Kronecker's canonical form and the qz algorithm. *Linear Algebra and its Applications*, 28:285–303, 1979.

[60] R. Winkler. Stochastic differential algebraic equations of index 1 and applications in circuit simulation. *Journal of computational and applied mathematics*, 157(2):477–505, 2003.

[61] X. Ye, W. Dong, P. Li, and S. Nassif. Maps: Multi-algorithm parallel circuit simulation. In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design*, pages 73–78, 2008.

[62] X. Zhang, Y. Liu, R. Coutts, and C.-K. Cheng. Power distribution network design optimization with on-die voltage-dependent leakage path. In *2013 IEEE 22nd Conference on Electrical Performance of Electronic Packaging and Systems*, pages 87–90. IEEE, 2013.

[63] Z. Zhu, H. Peng, C. K. Cheng, K. Rouz, M. Borah, and E. S. Kuh. Two-stage Newton-Raphson method for transistor-level simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(5):881–895, 2007.

[64] H. Zhuang, X. Wang, Q. Chen, P. Chen, and C. K. Cheng. From circuit theory, simulation to $spice^{Diego}$: A matrix exponential approach for time-domain analysis of large-scale circuits. *IEEE Circuits and Systems Magazine*, 16(2):16–34, 2016.

[65] H. Zhuang, S.-H. Weng, J.-H. Lin, and C. K. Cheng. MATEX: A distributed framework of transient simulation of power distribution networks. In *Proc. IEEE/ACM Design Autom. Conf.*, pages 43.3.1–6, 2014.

[66] H. Zhuang, W. Yu, S.-H. Weng, I. Kang, J.-H. Lin, X. Zhang, R. Coutts, and C. K. Cheng. Simulation algorithms with exponential integration for time-domain analysis of large-scale power delivery networks. *IEEE TCAD*, 35(10):1681–1694, 2016.