STATE SPACE ANALYSIS AND OPTIMIZATION OF MARX GENERATOR

BY

NICOLAS NEHME ANTOUN

B.E. Computer Engineering,
Lebanese American University,
Lebanon, June 2004

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico
Albuquerque, New Mexico

July, 2006

Dedication,

To my dear family

Acknowledgments

I would like to thank my advisor Professor Chaouki Abdallah for his continuous support through my graduate years as he put his trust in me and gave me the opportunity to develop both as a student and as a person. I want to express my appreciation to Professor Peter Dorato who was always available to offer his advice and his perspective of seeing things which always helped in better understanding the materials in question.

I wan to thank all my dear friends for their continuous support and help.

To my parents, whatever I say or do, I would never be able to express the appreciation and love I have for you. I want to thank you for everything that you did for me especially raising me in the best possible way during very hard circumstances.

STATE SPACE ANALYSIS AND OPTIMIZATION OF MARX GENERATOR

BY

NICOLAS NEHME ANTOUN

ABSTRACT OF THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico
Albuquerque, New Mexico

July, 2006

TITLE

STATE SPACE ANALYSIS AND OPTIMIZATION OF MARX GENERATOR

by

Nicolas Nehme Antoun

B.E., Computer Engineering, Lebanese American University, 2004
M.S., Electrical Engineering, University of New Mexico, 2006

ABSTRACT

We introduce in this thesis a set of procedures through which a user is given the ability to choose a particular desired output behavior from the circuit he is operating and obtain in return the corresponding set of design parameters that yield the requested output. We will demonstrate the applicability of these procedures on a Marx generator circuit. We proceed by introducing a general state space representation algorithm for any N stages Marx generator, then develop a time shifting algorithm that shifts the state trajectories of the system by the desired amount of time and apply a nonlinear Least-squares optimization algorithm to determine the set of design parameters.

Table of contents

List of Figures

List of Tables:

Chapter 1 Introduction

We offer, in this thesis, a circuit operating user with the capability of specifying his system’s output trajectory and provide him in return with the design parameters whose output best tracks the desired trajectory or reference model. As an application of this idea we will use a Marx pulse power generator circuit. Marx generators are based on charging a number of capacitors in parallel and discharging them in series [1]. Several circuit representation of a Marx generator exists depending on the manufacturer‘s design and components used. It was originally described by E. Marx in 1924 and is primarily used because of its ability to repetitively provide high bursts of voltages especially when the available voltage sources cannot provide the desired voltage levels [1]. Hence, a voltage source initially charges the capacitors which are then connected and discharged in series into the corresponding parasitic capacitors.

1.1 Motivation

The Marx generator is used for a wide range of applications in different research areas some of which are according to [2]:

- Generation of high power microwave using virtual cathode oscillator devices
- Lightning testing on cables and insulators.
- Material and dielectric testing.
- Breaking of raw diamonds in mineralogy.
- High voltage and magnetic pulser.
- High repetition rate high power CO_2 lasers.

- Generating EMP on parallel plate transmission lines.
- Bridge wire exploring.
- Electron injection into nuclear reactors.
- Electron accelerators.
- Kilo amp linear accelerators.
- Current injection and generation.
- Radiation generation for high voltage steep pulser.
- Flash x-ray generation.
- Pulsed electron generation.
- Short duration luminous flash for ultra high speed photography.
- Firing boxes for pyrotechnic substance reliability testing.
- Exploding unattended munitions.
- Nuclear electromagnetic pulse generator.
- Generation of plasma focusing.
- Generation of axial plasma for injection purposes.
- Remote de-programming of processors used in computers and other control circuitry.
- Educational demonstration of electrical pyrotechnics.

However, so far, no one has attempted a state space representation of an N stages Marx generator, and hence no one was able to exploit the simplifications induced by such a realization to be better design and control the generator. Researchers have attempted to improve the performance of Marx generators in terms of the electronics and hardware involved in putting the generator together as was done in [3] and stated in [2]. Hence, manufacturers deliver Marx generators with certain specifications and operational characteristics that the user has to adapt to.

1.2 Objective

The main objective of this thesis is to provide the end user with the ability to specify a desired behavioral performance from the output of his system, in our case from the Marx generator he is operating. Consequently, Marx generator models can be standardized, by providing their users with the ability of specifying the number of stages required for their application and the instant of time at which the spark should occur. This will eliminate the need to develop and produce a new generator for each application while freeing the end user from the constraints involved with some of the manufacturer's preset specifications.

1.3 Methodology

To achieve our objective, we decided to first generate an algorithm that determines the state space model of any N stages Marx generator. The next step was to choose a reference state space trajectory model, so we decided to start from a reference model that closely approximates the behavior of a Marx generator, but that is by no means ideal. Now, we want to provide the user with the ability to predefine the time at which the first spark is to occur. After providing the desired instant of time, we initialize a shifting factor parameter and develop an algorithm that exploits some of the state trajectory properties to move the state trajectories by the appropriate amount of time so that the spark occurs at the new, user specified time. After obtaining a new state trajectory reference model, we use a nonlinear least-squares optimization algorithm to determine the values of the design parameters, in our case the parasitic capacitor values, that best track the reference model. To show the effectiveness of this technique, we will present a comparison between the new simulated state trajectories and the corresponding model reference.

Hence, we will start Chapter 2 with the state space realization of an $N = 2$ stages Marx generator and then generalize the results to develop an algorithm that generates the state space model of for any N stages Marx generator. In Chapter 3, we explain the system discretization process required to successfully apply a nonlinear least-squares algorithm that we introduce in the same chapter and show its application to a reference state trajectory model. In the following Chapter 4, we present an algorithm that shifts the reference state trajectories by a specific shifting factor such that the spark at the $N+1$ parasitic capacitor occurs at a user-specified time. In the last chapter, Chapter 5, using two different shifting factors we apply the state-shifting algorithm of chapter 4 to our reference model, apply the nonlinear least-squares optimization algorithm to obtain the corresponding parasitic capacitor values and present the simulation results. Finally, we conclude this thesis by an overall conclusion summarizing the results that we obtained and proposing future work and applications.

1.4 Conclusions:

We have described in this chapter how a Marx pulse power generator works and listed some of the applications for which this generator is used. In addition, we have outlined the procedures that will be used to achieve our objective of providing the users with more control over their Marx generators.

Chapter 2 State Space Realization

We start this chapter by explaining the equations that govern the performance of any N stages Marx generator, then we present an $N = 2$ stages Marx generator, explain how it works and derive its corresponding state space model. Extrapolating from the state space representations of the $N = 2$ stages and $N = 4$ stages (presented in Appendix A) Marx generators, we develop an algorithm that automatically generates the state space realization any N stages Marx generator.

2.1 N-Stages Marx Generator General Characteristics:

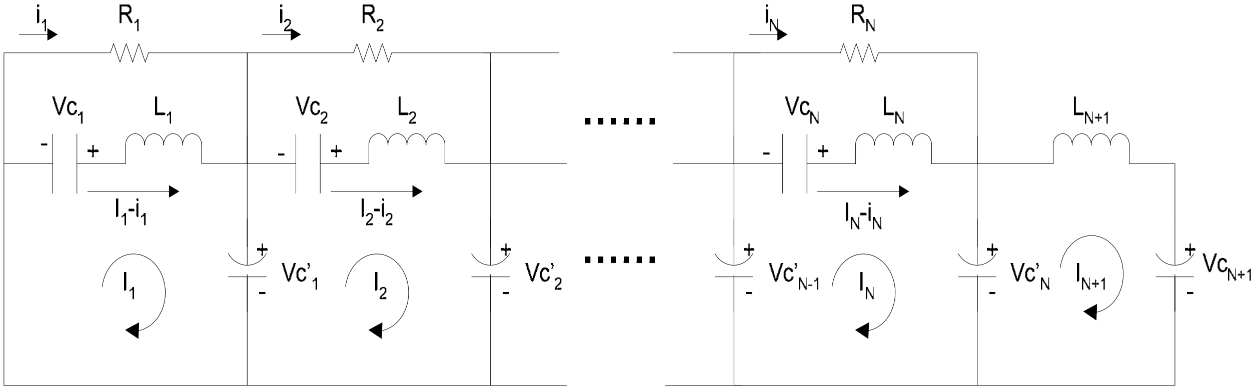

Figure 1 – N stages Marx Generator.

As explained in Chapter 1, an external voltage source simultaneously charges the $C_1, C_2, \cdots C_{N-1}, C_N$ capacitors. After charging these capacitors to the desired initial charge, the discharging process starts into the corresponding parasitic capacitors through their respective inductances and load resistances.

For all of the following N stages Marx generator models, the load resistances are such that $R_1 = R_2 = \cdots = R_N = 100{,}000\,\Omega$, thus the current i_j, for $1 \le j \le N$, across the j^{th}

resistor will be very small when compared to the corresponding I_j. Hence, we will assume from now on that the current across the j^{th} inductor is I_j instead of $I_j - i_j$. As a direct consequence of the previous assumption, during the discharging process of the N capacitors, the individual stages can be looked at as:

The first stage of the circuit

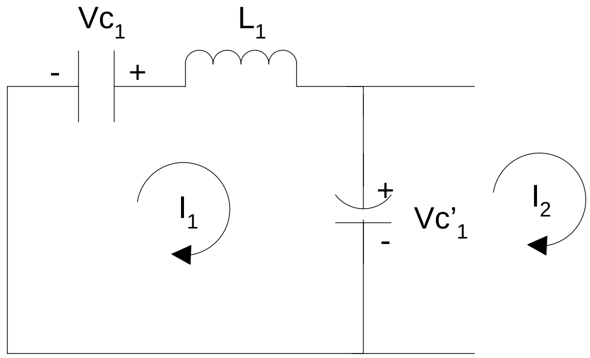

Figure 2- 1st stage discharge process

the governing voltage law is

$$Vc_1^{'} = Vc_1 - L_1 \frac{dI_1}{dt}$$

$$\Rightarrow Vc_1^{'} = Vc_1 + L_1 \frac{d^2 Vc_1}{dt^2}$$

The remaining N stages can be looked at as

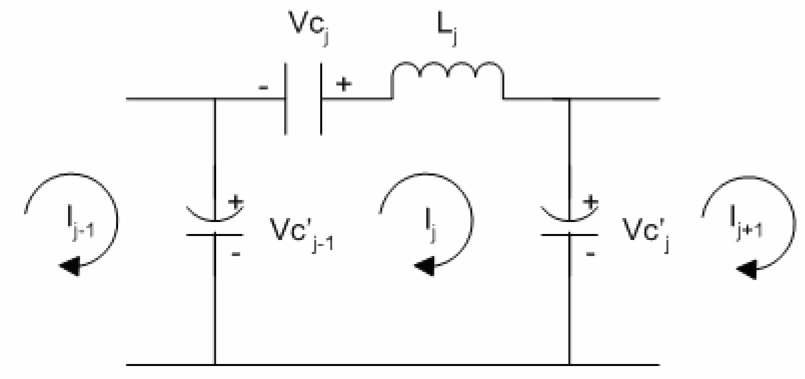

Figure 3- Generator's j^{th} stage discharging process

By examining Figure 2 we can write any j^{th} stage voltage equation as:

$$Vc_{j}^{'} = Vc_{j} - L_{j}\frac{dI_{j}}{dt} + Vc_{j-1}^{'} \text{ (1)}$$

where $I_{j} = -C_{j}\frac{dVc_{1}}{dt}$, hence equation (1) becomes

$$Vc_{j}^{'} = Vc_{j} + L_{j}C_{j}\frac{d^{2}Vc_{j}}{dt^{2}} + Vc_{j-1}^{'}$$

Note here that to write the previous two equations we assume the following:

1. at the first stage the stage capacitor C_1 discharges into $C_1^{'}$ while the next stage capacitor C_2 is not yet connected.
2. For the remaining N stages, the parasitic capacitor $C_{J-1}^{'}$ and C_j discharge into the corresponding $C_J^{'}$ while the next stages $C_{J+1}^{'}$

We now know that at the j^{th} stage the voltage equation is defined recursively in function of the previous parasitic capacitors voltages, therefore we can write a general voltage equation for any of the N stages:

$$Vc_j^{'} = Vc_j + L_j C_j \frac{d^2 Vc_j}{dt^2} + Vc_{j-1} + L_{j-1} C_{j-1} \frac{d^2 Vc_{j-1}}{dt^2} + \cdots + Vc_1 + L_1 C_1 \frac{d^2 Vc_1}{dt^2} \quad (2)$$

We can simplify the above equation if we have the following assumptions:

$L_j = L_{j-1} = \cdots = L_1 = L$

$C_j = C_{j-1} = \cdots = C_1 = C$

Hence, if the previous two constraints are satisfied equation (2) becomes

$$Vc_j^{'} = Vc_j + Vc_{j-1} + \cdots + Vc_1 + LC\left(\frac{d^2 Vc_j}{dt^2} + \frac{d^2 Vc_{j-1}}{dt^2} + \cdots + \frac{d^2 Vc_1}{dt^2} \right) \ (3)$$

Writing the voltage equation at the $N+1$parasitic capacitor, we obtain

$$\begin{aligned} Vc_{N+1}^{'} &= Vc_N^{'} - L_{N+1} C_{N+1} \frac{d^2 Vc_{N+1}^{'}}{dt^2} \\ \Rightarrow Vc_N^{'} &= Vc_{N+1}^{'} + L_{N+1} C_{N+1} \frac{d^2 Vc_{N+1}^{'}}{dt^2} \end{aligned} \quad (4)$$

If we replace j by N in equation (3), we obtain the following equality

$$Vc_N^{'} = Vc_N + Vc_{N-1} + \cdots + Vc_1 + LC\left(\frac{d^2 Vc_N}{dt^2} + \frac{d^2 Vc_{N-1}}{dt^2} + \cdots + \frac{d^2 Vc_1}{dt^2} \right)(5)$$

If we examine equations (4) and (5) in more details we notice that to have a consistent expression for the resonant frequency at the $N+1^{st}$ stage, the following equality should be satisfied

$$L_{N+1} C_{N+1} = LC \ \ (6)$$

When equality (6) is verified, the resonant radiant frequency of the N stage capacitors and the $N+1^{st}$ parasitic capacitor can be expressed by:

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Using the concept of conservation of energy, we know that all the initial energy stored in the $C_1, C_2, \cdots, C_N$ capacitors must be recovered at the $N+1^{th}$ stage, i.e. at C_{N+1}. Hence, knowing that the energy across a capacitor is $E = \frac{1}{2} CVc^2$, if all the currents and voltages across stage capacitors are zeros at t_f then the following equality must hold

$$\frac{1}{2} C'_{N+1} Vc'_{N+1}(t_f)^2 = \sum_1^N \frac{1}{2} C_j Vc_j(0)^2 \quad (7)$$

Where $Vc_j(0)$ represents the initial voltage to which the corresponding j^{th} capacitor was charged and $Vc'_{N+1}(t_f)$ is the total voltage discharged into the $N+1$ parasitic capacitor. Knowing that $C_N = C_{N-1} = \cdots = C_1 = C$ and $Vc_N = Vc_{N-1} = \cdots = Vc_1 = V_0$, the above equation (7) becomes

$$\frac{1}{2} C'_{N+1} Vc'_{N+1}(t_f)^2 = \frac{N}{2} CV_0{}^2 \quad (8)$$

the time t_f is such that

$$t_f = \frac{1}{2 f_0},$$

where $2\pi f_0 = \omega_0 \Rightarrow f_0 = \frac{1}{2\pi\sqrt{LC}}$.

The objective of Marx generators is to have $Vc'_{N+1}(t_f) = NV_0$, hence equation (8) becomes

$$\frac{1}{2} C'_{N+1} N^2 V_0^2 = \frac{N}{2} CV_0^2$$

this can only be achieved if

$$C'_{N+1} = N \cdot C$$

this in turn, according to equation (6), implies that

$$L_{N+1} = \frac{L}{N}$$

Note that these results are true for all the stages involved in any N stages Marx generator.

2.2 N=2-Stage Marx Generator State Space Representation

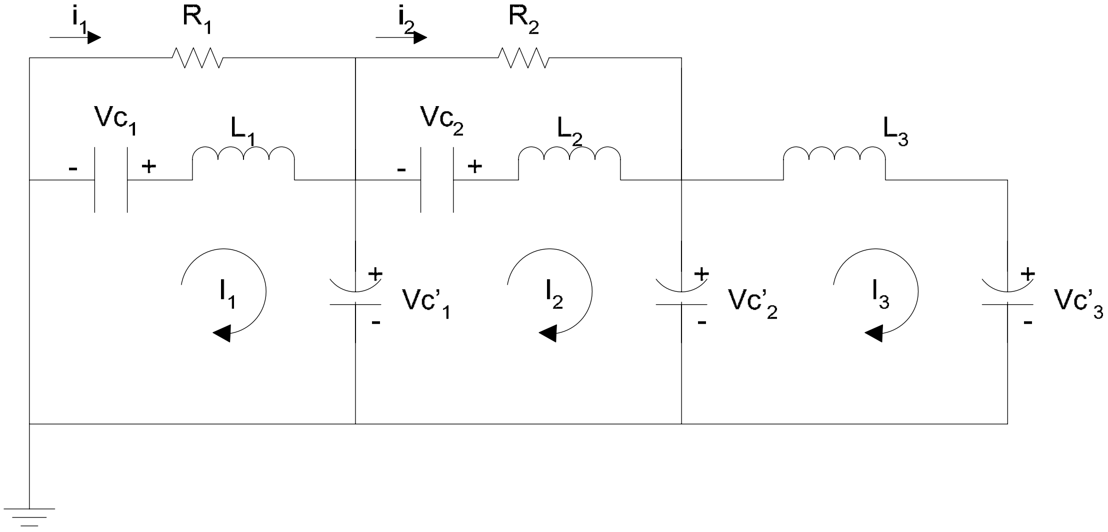

Figure 4- A 2-stages Marx Generator

The above figure displays an $N = 2$ stages Marx generator. We derive in this section its corresponding state space representation. Let

$$X_1 = Vc_1(t),\ X_2 = Vc_2(t),\ X_3 = Vc_1^{'}(t),\ X_4 = Vc_2^{'}(t),$$
$$X_5 = Vc_3^{'}(t),\ X_6 = I_1(t),\ X_7 = I_2(t),\ X_8 = I_3(t).$$

Using Graph analysis we can redraw the above circuit as a connection of branches, where tree branches represent voltage sources (in our case the capacitors), and links represent current sources (in our case the inductors) and resistors. The direction of the arrows is along the voltage drop in the case of a voltage source, or along the current in the case of a current source [4]. The corresponding graph representation is therefore:

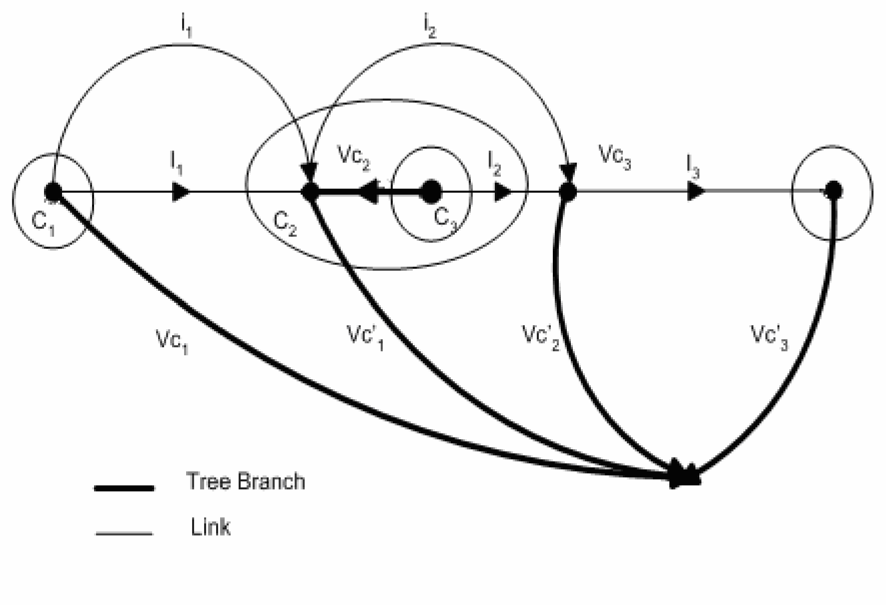

Figure 5- Graph of N=2-stage Marx Generator

Having chosen the states to be voltages across capacitors and currents across inductors we follow these two simple rules stated in [4]:

1. Write KCL for every fundamental cut set (i.e. one tree branch and a number of links) in the network formed by each capacitor in the tree.
2. Write KVL for every fundamental loop (i.e. one link and a number of tree branches) in the network formed by each inductor in the co-tree (complement of a tree).

Cut set C_1: $C_1 \frac{dVc_1}{dt} + i_1 + I_1 = 0 \Rightarrow C_1 \dot{X}_1 + i_1 + X_6 = 0$ (9)

Cut set C_2: $-i_1 - I_1 + C_1^{'} \frac{dVc_1^{'}}{dt} + i_2 + I_2 = 0 \Rightarrow -i_1 - X_6 + C_1^{'} \dot{X}_3 + i_2 + X_7 = 0$ (10)

Cut set C₃: $C_2 \frac{dVc_2}{dt} + I_2 = 0 \Rightarrow C_2 \dot{X}_2 + X_7 = 0 \Rightarrow \dot{X}_2 = -\frac{1}{C_2} X_7$ (11)

Cut set C₄: $-i_2 - I_2 + C_2^{'} \frac{dVc_2^{'}}{dt} + I_3 = 0 \Rightarrow -i_2 - X_7 + C_2^{'} \dot{X}_4 + X_8 = 0$ (12)

Cut set C₅: $C_3^{'} \frac{dVc_3^{'}}{dt} - I_3 = 0 \Rightarrow C_3^{'} \dot{X}_5 - X_8 = 0 \Rightarrow \dot{X}_5 = -\frac{1}{C_3^{'}} X_8$ (13)

Loop 1 ($I_1 \rightarrow Vc_1^{'} \rightarrow Vc_1$ **):**

$L_1 \frac{dI_1}{dt} + Vc_1^{'} - Vc_1 = 0 \Rightarrow L_1 \dot{X}_6 + X_3 - X_1 = 0 \Rightarrow \dot{X}_6 = \frac{1}{L_1} X_1 - \frac{1}{L_1} X_3$ (14)

Loop 2 ($I_2 \rightarrow Vc_2^{'} \rightarrow Vc_1^{'} \rightarrow Vc_2$ **):**

$L_2 \frac{dI_2}{dt} + Vc_2^{'} - Vc_1^{'} - Vc_2 = 0 \Rightarrow L_2 \dot{X}_7 + X_4 - X_3 - X_2 = 0 \Rightarrow$
$\dot{X}_7 = \frac{1}{L_2} X_2 + \frac{1}{L_2} X_3 - \frac{1}{L_2} X_4$ (15)

Loop 3 ($I_3 \rightarrow Vc_3^{'} \rightarrow Vc_2^{'}$ **):**

$L_3 \frac{dI_3}{dt} + Vc_3^{'} - Vc_2^{'} = 0 \Rightarrow L_3 \dot{X}_8 + X_5 - X_4 = 0 \Rightarrow \dot{X}_8 = \frac{1}{L_3} X_4 - \frac{1}{L_3} X_5$ (16)

Eliminating i_1, i_2 :

Loop 6 ($i_1 \rightarrow Vc_1^{'} \rightarrow Vc_1$ **):**

$R_1 i_1 + Vc_1^{'} - Vc_1 = 0 \Rightarrow R_1 i_1 + X_3 - X_1 = 0 \Rightarrow i_1 = \frac{1}{R_1}(X_1 - X_3)$ (17)

Loop 7 ($i_2 \rightarrow Vc_2^{'} \rightarrow Vc_1^{'}$ **):**

$R_2 i_2 + Vc_2^{'} - Vc_1^{'} = 0 \Rightarrow R_2 i_2 + X_4 - X_3 = 0 \Rightarrow i_2 = \frac{1}{R_2}(X_3 - X_4)$ (18)

Replacing (17) in (9) we obtain:

$$C_1\dot{X}_1 + \frac{1}{R_1}X_1 - \frac{1}{R_1}X_3 + X_6 = 0 \Rightarrow \dot{X}_1 = -\frac{1}{R_1C_1}X_1 + \frac{1}{R_1C_1}X_3 - \frac{1}{C_1}X_6 \ \ (9)$$

Replacing (17) and (18) in (11) we obtain:

$$-\frac{1}{R_1}X_1 + \frac{1}{R_1}X_3 - X_6 + C_1^{'}\dot{X}_3 + \frac{1}{R_2}X_3 - \frac{1}{R_2}X_4 + X_7 = 0$$

$$\Rightarrow -\frac{1}{R_1}X_1 + \frac{R_1 + R_2}{R_1R_2}X_3 - \frac{1}{R_2}X_4 - X_6 + X_7 + C_1^{'}\dot{X}_3 = 0$$

$$\dot{X}_3 = \frac{1}{R_1C_1^{'}}X_1 - \frac{R_1+R_2}{R_1R_2C_1^{'}}X_3 + \frac{1}{R_2C_1^{'}}X_4 + \frac{1}{C_1^{'}}X_6 - \frac{1}{C_1^{'}}X_7 \ \ (11)$$

Replacing (18) in (12) we obtain:

$$-X_7 - \frac{1}{R_2}X_3 + \frac{1}{R_2}X_4 + X_8 + C_2^{'}\dot{X}_4 = 0 \Rightarrow$$

$$\dot{X}_4 = \frac{1}{R_2C_2^{'}}X_3 - \frac{1}{R_2C_2^{'}}X_4 + \frac{1}{C_2^{'}}X_7 - \frac{1}{C_2^{'}}X_8 \ \ (12)$$

Now we have the following set of equations that best describe the state space model of an N=2-stage Marx generator:

$$\dot{X}_1 = -\frac{1}{R_1C_1}X_1 + \frac{1}{R_1C_1}X_3 - \frac{1}{C_1}X_6 \ \ (9)$$

$$\dot{X}_2 = -\frac{1}{C_2}X_7 \ \ (10)$$

$$\dot{X}_3 = \frac{1}{R_1C_1^{'}}X_1 - \frac{R_1+R_2}{R_1R_2C_1^{'}}X_3 + \frac{1}{R_2C_1^{'}}X_4 + \frac{1}{C_1^{'}}X_6 - \frac{1}{C_1^{'}}X_7 \ \ (11)$$

$$\dot{X}_4 = \frac{1}{R_2C_2^{'}}X_3 - \frac{1}{R_2C_2^{'}}X_4 + \frac{1}{C_2^{'}}X_7 - \frac{1}{C_2^{'}}X_8 \ \ (12)$$

$\dot{X}_5 = -\frac{1}{C_3^{'}} X_8$ (13)

$\dot{X}_6 = \frac{1}{L_1} X_1 - \frac{1}{L_1} X_3$ (14)

$\dot{X}_7 = \frac{1}{L_2} X_2 + \frac{1}{L_2} X_3 - \frac{1}{L_2} X_4$ (15)

$\dot{X}_8 = \frac{1}{L_3} X_4 - \frac{1}{L_3} X_5$ (16)

Hence, we can now write our state space representation in the following form:

$$\dot{X} = {}^2M \cdot X \ ,$$

Where ${}^2M = \begin{bmatrix} {}^2M_{11} & {}^2M_{12} \\ {}^2M_{21} & {}^2M_{22} \end{bmatrix}$ is an 8×8 matrix and X is a 1x8 column vector.

${}^2M_{11}$ is a 5×5 matrix with the following structure:

$-\frac{1}{R_1 C_1}$	0	$\frac{1}{R_1 C_1}$	0	0
0	0	0	0	0
$-\frac{1}{R_1 C_1^{'}}$	0	$-\frac{R_1 + R_2}{R_1 R_2 C_1^{'}}$	$\frac{1}{R_2 C_1^{'}}$	0
0	0	$\frac{1}{R_2 C_2^{'}}$	$-\frac{1}{R_2 C_2^{'}}$	0
0	0	0	0	0

Table 1- ${}^2M_{11}$ matrix

${}^2M_{12}$ is a 5×3 matrix with the following structure:

$-\frac{1}{C_1}$	0	0
0	$-\frac{1}{C_2}$	0
$\frac{1}{C_1^{'}}$	$-\frac{1}{C_1^{'}}$	0
0	$\frac{1}{C_2^{'}}$	$-\frac{1}{C_2^{'}}$
0	0	$\frac{1}{C_3^{'}}$

Table 2- ${}^{2}M_{12}$ matrix

${}^{2}M_{21}$ is a 3×5 matrix with the following structure:

$\frac{1}{L_1}$	0	$-\frac{1}{L_1}$	0	0
0	$\frac{1}{L_2}$	$\frac{1}{L_2}$	$-\frac{1}{L_2}$	0
0	0	0	$\frac{1}{L_3}$	$-\frac{1}{L_3}$

Table 3- ${}^{2}M_{21}$ matrix

${}^{2}M_{22}$ is a 3×3 matrix with the following structure:

0	0	0
0	0	0
0	0	0

Table 4- ${}^{2}M_{22}$ matrix

Please note for this $N = 2$ stages Marx generator structure the following parameters were used:

$C = C_1 = C_2 = 1\,\mathrm{F}$,

$L = L_1 = L_2 = 1\,\mathrm{H}$,

$R = R_1 = R_2 = 100\,\mathrm{K}\,\Omega$,

$L_3 = N \times L = 2 \times 1 = 2\mathrm{H}$,

$C_1^{'} = 0.1578005\,\mathrm{F}, C_2^{'} = 0.28165\,\mathrm{F}$,

$C_5^{'} = \frac{C}{N} = \frac{1}{2}\mathrm{F}$

with resonant frequency, $\omega_0 = \frac{1}{\sqrt{LC}} = 1\,\mathrm{rad/sec}$.

During the simulation of this system we initially set $I_{1,2,3} = 0\mathrm{A}$, $Vc^{'}_{1,2,3} = 0\mathrm{V}$ and $Vc_{1,2} = 1\mathrm{V}$, which represents the initial voltage to which the C_1 and C_2 capacitors were charged. The following graph represents the voltage trajectories across the parasitic capacitors:

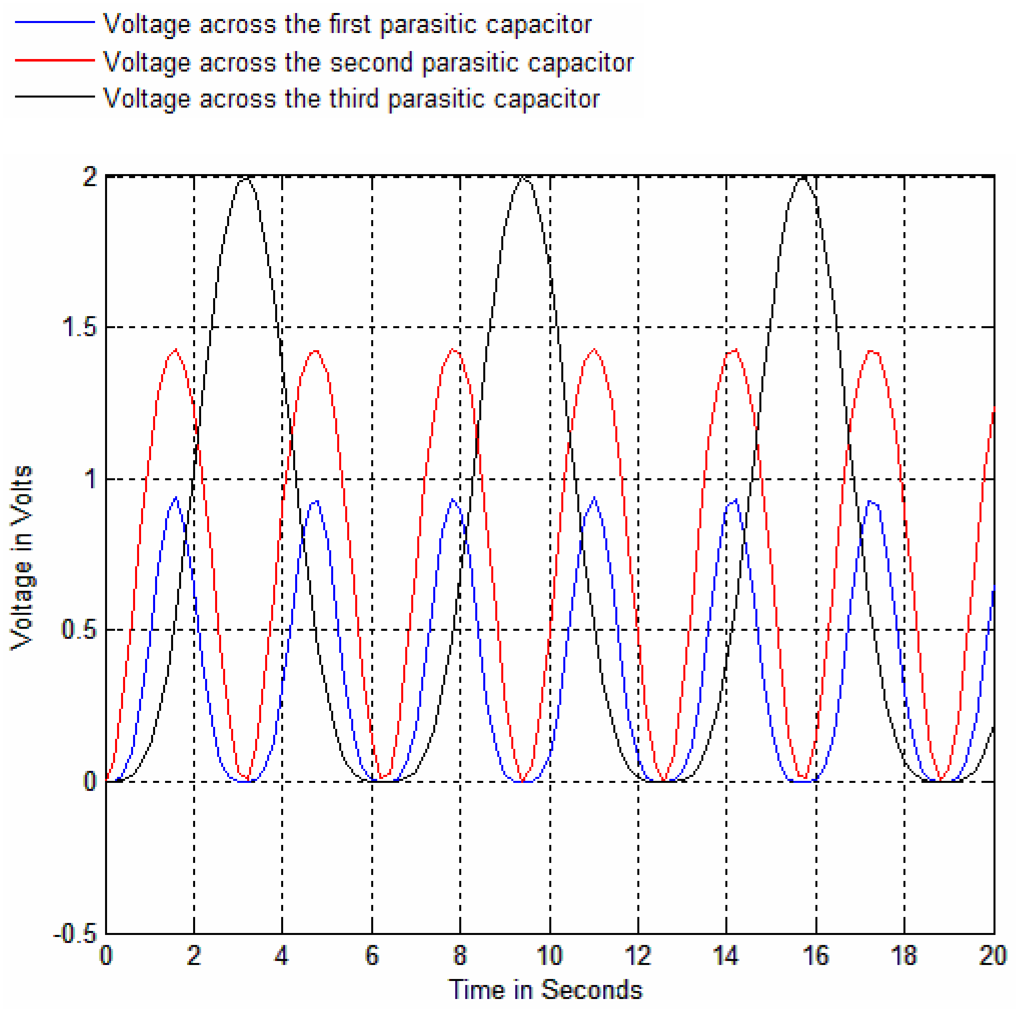

Figure 6- State trajectory representing the voltage across the parasitic capacitors of a 2-stage Marx generator

Clearly when looking at Figure 6 one can notice that when the voltage across the first two parasitic capacitors is zero the voltage across the third parasitic capacitor $Vc_3^{'}(t_f) = N \cdot Vc_{1,2}(0) = 2 \cdot 1 = 2\,\mathrm{V}$, where $t_f = \frac{2\pi\sqrt{LC}}{2} = \frac{2\pi\sqrt{1}}{2} = \pi \approx 3.142$ seconds. Note here that at t_f the currents across the inductors and voltages across the stage capacitors are also zero.

Please refer to Appendix A for the derivation of the state space model of an $N = 4$ stages Marx generator.

2.3 N-Stage Marx Generator general structure state space model

Before delving into the development of the algorithm, it is now clear that the number of states for an N-stage Marx generator is $S = 3N + 2$. This formula was deduced from the fact that for 2 stages the number of states is 8 and for 4 stages the number of states is 14 (as can be seen in Appendix A).

Going from the state space models of $\mathrm{N} = 2$ and 4 stages Marx generator we can generalize and extrapolate the structure of the ${}^{N}M$ matrix for any number of stages N as follows:

- ${}^{N}M$ can be divided into four blocks as follows:

$${}^{N}M = \begin{bmatrix} {}^{N}M_{11} & {}^{N}M_{12} \\ {}^{N}M_{21} & {}^{N}M_{22} \end{bmatrix}$$

- ${}^{N}M_{11}$ is a $(2N+1)\times(2N+1)$ matrix with the following structure

    - $\begin{bmatrix} m_{1,1}\cdots\cdots\cdots\cdots\cdots m_{1,2N+1} \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ m_{2N+1,1}\cdots\cdots\cdots m_{2N+1,2N+1} \end{bmatrix}$, where

    - $m_{1,1} = -m_{1,N} = -\dfrac{1}{R_1C_1}, \quad m_{1,2\to N-1} = m_{1,N+2\to 2N+1} = 0.$

- $m_{2\to N,1\to 2N+1} = 0$

- $m_{N+1,1} = \frac{1}{R_1C_1^{'}}$, $m_{N+1,2\to N} = m_{N+1,N+3\to 2N+1} = 0$, $m_{N+1,N+1} = -\frac{R_1+R_2}{R_1R_2C_1^{'}}$,

  $m_{N+1,N+2} = \frac{1}{R_2C_1^{'}}$.

- $m_{N+i,1\to N} = m_{N+i,N+2i\to 2N+1} = 0$, $m_{N+i,N+(i-1)} = \frac{1}{R_iC_i^{'}}$, $m_{N+i,N+i} = -\frac{R_i+R_{i+1}}{R_iR_{i+1}C_i^{'}}$,

  $m_{N+i,N+(i+1)} = \frac{1}{R_{i+1}C_i^{'}}$, for $i = 2 \to N$.

- $m_{2N+1,1\to 2N+1} = 0$

➢ ${}^{N}M_{12}$ is a $(2N+1)\times(N+1)$ matrix with the following structure

- $$\begin{bmatrix} m_{1,1}\cdots\cdots\cdots\cdots\cdots m_{1,N+1} \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad \vdots \\ m_{2N+1,1}\cdots\cdots\cdots m_{2N+1,N+1} \end{bmatrix}$$

- $m_{i,i} = -\frac{1}{C_i}$, for $1 \le i \le N$ and $m_{i,j} = 0$, for $i \ne j$ such that $1 \le i \le N$ and

  $1 \le j \le N+1$.

- $m_{i,i} = -m_{i,i+1} = \frac{1}{C_j^{'}}$ for $N+1 \le i \le 2N$ and $j = 1 \to N$.

- $m_{2N+1,1\to N} = 0$ and $m_{2N+1,N+1} = 0$.

➢ ${}^{N}M_{21}$ is an $(N+1)\times(2N+1)$ matrix with the following structure

- $$\begin{bmatrix} m_{1,1} \cdots\cdots\cdots\cdots\cdots m_{1,N+1} \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ m_{2N+1,1} \cdots\cdots\cdots m_{2N+1,N+1} \end{bmatrix}$$
- $m_{i,i} = \frac{1}{L_i}$ for $1 \le i \le N$ and $m_{i,j} = 0$, for $i \ne j$ such that $1 \le i \le N$ and $1 \le j \le N$. $m_{N+1,j} = 0$ for $1 \le j \le N$.
- $m_{1,N+1} = \frac{1}{L_1}$ and $m_{1,j} = 0$ for $N + 2 \le j \le 2N + 1$.
- $m_{i,i} = -m_{i,i+1} = \frac{1}{L_j}$ for $2 \le i \le N + 1$and $j = 1 \to N + 1$, $m_{i,j} = 0$ for $i \ne j$ such that $2 \le i \le N + 1$ and $N + 1 \le j \le 2N + 1$.

➢ ${}^{N}M_{22}$ is an $(N+1)\times(N+1)$ matrix with the following structure

- $$\begin{bmatrix} m_{1,1} \cdots\cdots\cdots\cdots\cdots m_{1,N+1} \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ m_{N+1,1} \cdots\cdots\cdots m_{N+1,N+1} \end{bmatrix}$$, where all entries $m_{i,j} = 0$ for all $1 \le i \le N + 1$ and $1 \le j \le N + 1$.

As a direct application of this algorithm, we present in Appendix A the state space model of an $N = 8$ stages Marx generator.

2.4 Conclusions

This chapter demonstrated the use of graph theory to determine the state space realization of an $N = 2$ stages Marx generator. By carefully inspecting the state space models of an $N = 2$ and $N = 4$ stages Marx generators, we developed an algorithm that generates the state space model of any N stages Marx generator. To this extent, we have also included in Appendix A the state space model of an $N = 8$ stages Marx generator using the algorithm of sectionXX.

Chapter 3 System Discretization and Optimization

We start this chapter with the procedures involved in dicretizing the continuous time model obtained in the previous chapter. Then we explain the least-squares nonlinear optimization algorithm and use it in conjunction with the generator's discrete time model to determine the values of the parasitic capacitors that will best track a reference state trajectory model. Please note that from now on, we will be using an $N = 4$ stages Marx generator to explain and demonstrate our work and results.

3.1 System discretization

Using the state space generation algorithm of 6Chapter 2, we can now determine the matrix ${}^{N}M$ such that

$$\dot{X}(t) = {}^{N}M \cdot X(t)$$

In particular for $N = 4$, we will have the following continuous time state space model:

$$\dot{X}(t) = {}^{4}M \cdot X(t)$$

where ${}^{4}M$, is the 14×14 matrix presented in Chapter 2, and $X(t)$ is a 14×1 vector containing 14 states of the $N = 4$ stages Marx generator.

The solution to the above equation is given by

$$X(t) = \exp({}^{4}M \cdot t) \cdot X(0)$$

Before applying this optimization scheme, the model was discretized using a time step of $h = 0.002$ seconds, such that the discrete time model representation is of the form

$$X(k+1) = G \cdot X(k)$$
$$\Rightarrow X(k) = G^k \cdot X(0)$$

where,

$$G = I + {}^4Mh + \frac{\left({}^4Mh\right)^2}{2!} + H.O.T.,$$

$H.O.T.$ are the higher order terms that cancel out as the power of h increases i.e. as $h \uparrow H.O.T \to 0$ and I is an identity matrix of size equal to the size of the ${}^4M$ matrix, that is 14×14.

3.2 Optimization overview:

Optimization is an approach used to determine the optimal value of a set of design parameters such that it minimizes or maximizes a defined objective function. Additional constraints could be defined as lower and upper bounds on the parameters and inequality or equality constraints on functions of the parameters. In the case where the objective function and the constraint equations are linear functions of the design variables then the problem can be solved as a Linear Programming problem. On the other hand, Nonlinear Programming problem, where the objective function and the constraint equations are nonlinear in the design variables, the solution is obtained using an iterative process during which a new direction of search is calculated at each of the iterations [5].

After determining the state space model that best describes the N=4 stages Marx generator, the objective became to find what values of the parasitic capacitors yield the desired state behavior. Based on our reference model and as a proof of concept, we will try to track the states' behavior using a nonlinear least-squares optimization algorithm and show the effectiveness of this approach when used for our application.

3.3 Optimization algorithm

The optimization scheme is based on minimizing a set of objective functions simultaneously. These functions are stored in a vector of functions called F . To generate the entries in F, we choose an interval of time for which we want to track the reference state trajectories. The general structure of the equations in F is

$$X_r(i+1, j) - X_d(k+1, j),$$

Where X_r is, a 10001×14 matrix, obtained from $\dot{X}_r = {}^4M \cdot X_r$, using a time vector spanning the interval $0 \le T \le 20$ seconds, of size 1×10001, a step size $h = 0.002$, and the following initial conditions:

$$X_r(0) = [3,3,3,3,0,0,0,0,0,0,0,0]$$

The non-zero entries represent the initial voltages up to which the the capacitors C_1, C_2, C_3, C_4 were charged.

Thus, $X_r(i+1, j)$ represents the value of the j^{th} entry in matrix X_r at instant of time $T(i+1)$.

$X_d(k+1,:) = G \cdot X_d(k,:)'$, is a 14×1 vector, G is a 14×14 matrix, and $X_d(k,:)'$ a 14×1 vector containing the values of the 14 states at time instant $k+1$. Therefore, $X_d(k+1, j)$ corresponds to the value of the j^{th} state at time step $k+1$.

The algorithm we use is a conjunction of line search procedures and a quasi-Newton algorithm; the Levenberg-Marquardt method. The function that is minimized is considered as a sum of squares:

$$\min_{U \in \Re^N} f(U) = \frac{1}{2}\|F(U)\|_2^2 = \frac{1}{2}\sum_p F_p(U)^2 \qquad (1)$$

where $F_p(U) = X_r(i+1, j) - X_d(k+1, j)$.

More accurately, according to [5], we are performing a nonlinear parameter estimation to fit a model function to data generated in X_r. $F(U)$, has the following structure:

$$F(U) = \begin{bmatrix} F_1(U) \\ F_2(U) \\ \vdots \\ F_n(U) \end{bmatrix}$$

Please note that in the following, N represents the number of stages in the Marx generator and therefore, for the current case we are studying, $N = 4$. $F(U)$ has the following properties:

- $U^T = [C_1^{'} \quad C_2^{'} \quad \cdots\cdots \quad C_N^{'}]$ is an $1 \times N$ vector containing the values of the N design parameters to be determined.
- $J(U)$ is a $n \times N$ Jacobian matrix of $F(U)$.
- $G(U)$ the $N \times 1$ gradient vector of $f(U)$.
- $H(U)$ the $N \times N$ Hessian matrix of $f(U)$.

3.3.1 Quasi-Newton Methods:

Quasi-Newton methods are the most popular methods that use gradient information. These methods formulate the problem as a quadratic problem represented by:

$$f(U) = \min_U \left(\frac{1}{2} U^T H\, U + c^T U + b \right)$$

where the Hessian matrix, H , is a positive definite symmetric matrix, c is a constant vector, and b is a constant. The optimal solution of this problem occurs when the partial derivatives of $f(U)$ go to zero [5]:

$$\nabla f(U^*) = H\ U^* + c = 0$$

Where, according to [6], $\dfrac{d\left(\frac{1}{2} U^T H\, U \right)}{dU} = H\ U$, $\dfrac{d\left(c^T U\right)}{dU} = c$ and U^* represents the optimal solution point.

The advantage of quasi-Newton methods over Newton methods is the way the Hessian matrix H is calculated. Newton-type methods calculate H directly and proceed in a direction of descent to iteratively locate the minimum, which introduces computational complexities that comes from the numerical generation of H . On the other hand, Quasi-Newton methods use values of $f(U)$ and $\nabla f(U)$ to construct curvature information and generate an approximation of what H should be using the appropriate updating technique. The most efficient updating method, that proved to guarantee fast convergence rate and global convergence under the right conditions [13], has been developed by Broyden [7], Fletcher [8], Goldfarb [9], and Shanno [10], hence its name BFGS:

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k}$$

where

$$\begin{aligned} s_k &= U_{k+1} - U_k \\ q_k &= \nabla f(U_{k+1}) - \nabla f(U_k) \end{aligned}$$

H_0, the starting point of the updating technique, can be set to any positive definite symmetric matrix in particular to the identity matrix. Instead of calculating the inverse of the Hessian H^{-1}, the DFP formula, derived by Davidon [11], Fletcher and Powell [12], is used. The DFP formula uses the same formula as BFGS however while substituting q_k by s_k:

$$H_{k+1}^{-1} = H_k^{-1} + \frac{s_k s_k^T}{s_k^T q_k} - \frac{H_k^T q_k^T q_k H_k}{q_k^T H_k q_k}$$

Similarly, the gradients of the objective function entries are obtained using a numerical differentiation method via finite differences based on changing the value of each of the design parameters and calculating the corresponding rate of change in the objective function [5].

The direction of search for each iteration can be calculated as follows:

$$d_k = -H_k^{-1} \cdot \nabla f(U_k)$$

3.3.2 Line Search:

Line search is the search method used by the Levenberg-Marquardt algorithm to find the search direction towards the minimum of the objective function. At each step of the main algorithm, the line-search method generates a new set of design variables for the next iteration:

$$U_{k+1} = U_k + \alpha^* d_k$$

where U_k denotes the current iterate, d_k is the search direction and α^* is a scalar step length parameter.

The line search method attempts to decrease the objective function along the line $U_k + \alpha^* d_k$ by continuously minimizing the objective function. The line search procedure consists of two phases:

1. The *bracketing* phase: corresponding to an interval specifying the range of values α to be tried-out along the line $U_{k+1} = U_k + \alpha^* d_k$ to be searched such that $f(U_{k+1}) < f(U_k)$.
2. The *sectioning* phase: dividing the bracket determined in the bracketing phase into subintervals on which the minimum of the objective function is approximated by polynomial interpolation.

The resulting step length α satisfies the Wolfe conditions:

$$\begin{gathered}(x_k + \alpha d_k) \leq f(x_k) + c_1 \alpha \nabla f_k^T d_k \\ \nabla f(x_k + \alpha d_k)^T d_k \geq c_2 \alpha \nabla f_k^T d_k\end{gathered}$$

where c_1 and c_2 are constants with $0 < c_1 < c_2 < 1$.

These conditions guarantee that we will be using the largest value of α that decreases the objective function.

3.3.3 Quasi-Newton Implementation:

The quasi-Newton method used consists of two parts:

- Determining the direction of search from the updated Hessian matrix using the BFGS formula;

By looking at the Hessian update formula again:

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k}$$

where

$$s_k = U_{k+1} - U_k$$
$$q_k = \nabla f(U_{k+1}) - \nabla f(U_k)$$

The sign of H_{k+1} is dominated by the term $q_k^T s_k$:

$$q_k^T s_k = \left(\nabla f(U_{k+1}) - \nabla f(U_k)\right)^T \cdot \left(U_{k+1} - U_k\right)$$
$$q_k^T s_k = \left(\nabla f(U_{k+1})^T - \nabla f(U_k)^T\right) \cdot \left(U_k + \alpha_k d_k - U_k\right)$$
$$q_k^T s_k = \left(\nabla f(U_{k+1})^T - \nabla f(U_k)^T\right) \cdot \left(\alpha_k d_k\right)$$

Knowing that α_k is a constant 1×1 term, we obtain the following expression for $q_k^T s_k$:

$$q_k^T s_k = \alpha_k \left(\nabla f(U_{k+1})^T - \nabla f(U_k)^T\right) \cdot d_k$$

We know that $f(U_k) = \frac{1}{2} \sum_p F_p (U_k)^2$, then it is guaranteed that $f(U_k) > 0$ and $\nabla f(U_k) > 0$. Consequently, $-f(U_k) < 0$ and $-\nabla f(U_k) < 0 \rightarrow -\nabla f(U_k)^T < 0$. From section XX $d_k = -H_k^{-1} \cdot \nabla f(x_k)$, hence $d_k < 0$:

$$-\nabla f(U_k)^T d_k > 0$$

$\Rightarrow -\alpha \nabla f(U_k)^T d_k$ is guaranteed to be positive negative. However, the term $-\alpha \nabla f(U_{k+1})^T d_k$ can still be negative. This is where the design of α_k comes into play to guarantee that $q_k^T s_k$ is positive definite by guaranteeing:

$$-\alpha \nabla f(U_{k+1})^T d_k + -\alpha \nabla f(U_k)^T d_k > 0$$
$$\Rightarrow -\alpha \nabla f(U_{k+1})^T d_k > -\alpha \nabla f(U_k)^T d_k$$

Hence, the Hessian matrix at each iteration is guaranteed to be positive definite so that the direction of search d is always negative and hence in a descent direction.

Consequently, a small step α in the same direction of d will decrease the magnitude of the objective function.

- The line search procedures;

At each iteration, before a Hessian update is made, the following condition must be checked:

$$f(U_{k+1}) < f(U_k)$$

where $U_{k+1} = U_k + \alpha_k d$.

If U_{k+1} does not satisfy the condition above then α_k is reduced to form a new iteration step α_{k+1}. The usual reduction method is a bisection method that involves halving the current value of α until a reduction in $f(U)$ is observed.

When a U that satisfies the condition above is found, the Hessian matrix is updated if the term $q_k^T s_k$ is positive. If $q_k^T s_k$ is not positive then further cubic interpolation is performed such that a valid U_{k+1} is found that satisfies the following conditions:

$$f(U_{k+1}) < f(U_k)$$

$$\nabla f(U_{k+1})^T d \text{ is small enough such that } q_k^T s_k > 0$$

As U_k approaches the solution point, the procedure goes back to using an α_k that is close to unity.

3.4 Application to the reference model:

That being said, we applied the nonlinear least-squares optimization algorithm explained above to the reference state matrix X_r, displayed in Figure 36 of Appendix A. The results turned out as follows:

$$C_1^{'} = 0.0359876\,,\ C_2^{'} = 0.06721539\,,\ C_3^{'} = 0.02362871\,,\ C_4^{'} = 0.12467976\,.$$

Looking at the parasitic capacitor values used for the reference model

$${}^{r}C_1^{'} = 0.0359864\,,\ {}^{r}C_2^{'} = 0.067215\,,\ {}^{r}C_3^{'} = 0.02362875\,,\ {}^{r}C_4^{'} = 0.12467875$$

Hence, we have the following differences between the reference model parasitic capacitors values and the one obtained from the least-squares optimization algorithm:

$${}^{r}C_1^{'} - C_1^{'} = -1.2\times10^{-6}\ ,\ {}^{r}C_2^{'} - C_2^{'} = -3.9\times10^{-7}\ ,\ {}^{r}C_3^{'} - C_3^{'} = 4\times10^{-8}\ ,$$

$${}^{r}C_4^{'} - C_4^{'} = -1.01\times10^{-6}\,.$$

Which shows that the error between the two sets of parasitic capacitors is relatively small with an average of $-2.56\times10\wedge-6$.

By using the values for the parasitic capacitors obtained from the optimization algorithm we obtained the following state trajectories stored in X_{opt}:

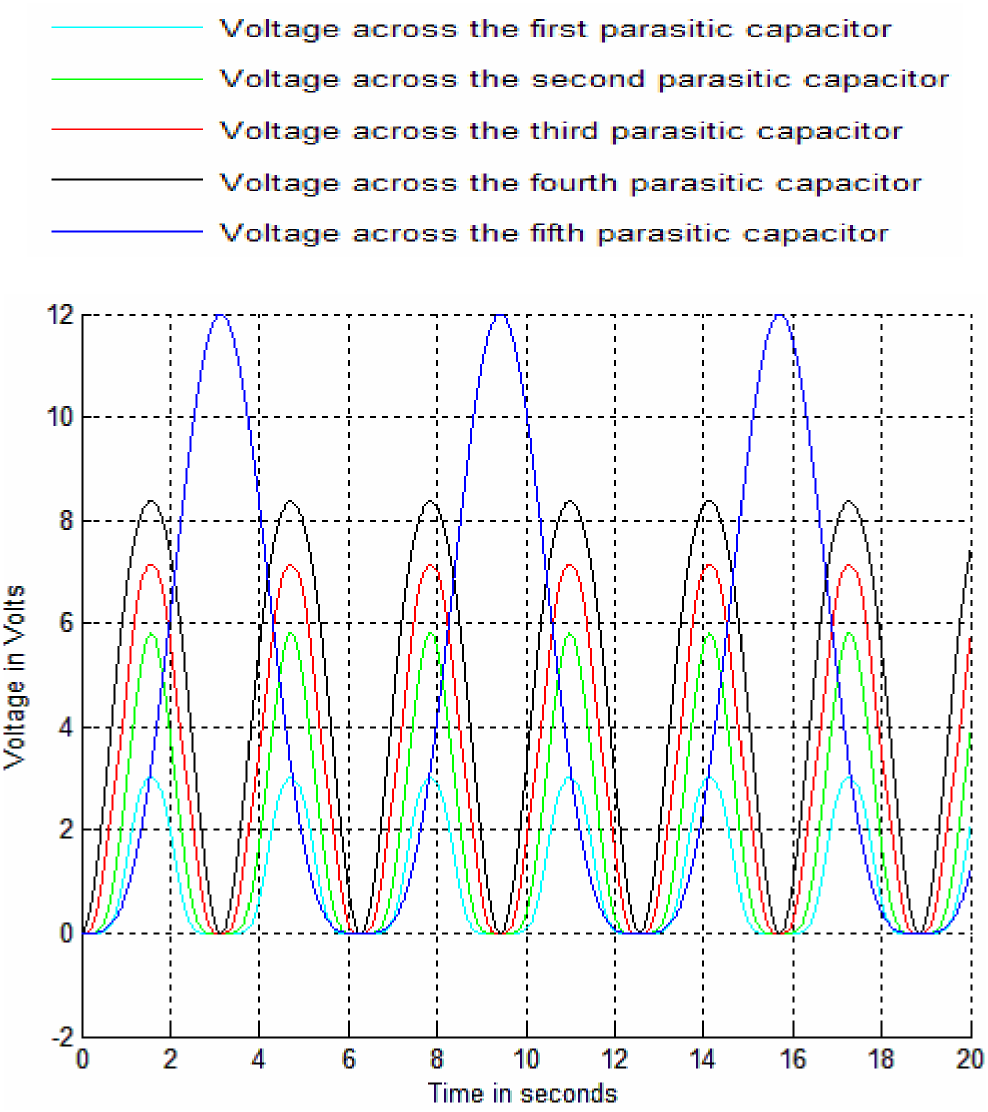

Figure 7- State trajectories X_{opt} obtained using the parasitic capacitor values from the optimization algorithm

The relative error between the X_r and X_{opt}, calculated using the following formula:

$$err(i,j) = \frac{X_r(i,j) - X_{opt}(i,j)}{X_r(i,j)}, \text{ for } 1 \le i \le 10001 \text{ and } 1 \le i \le 14$$

Looks as follows:

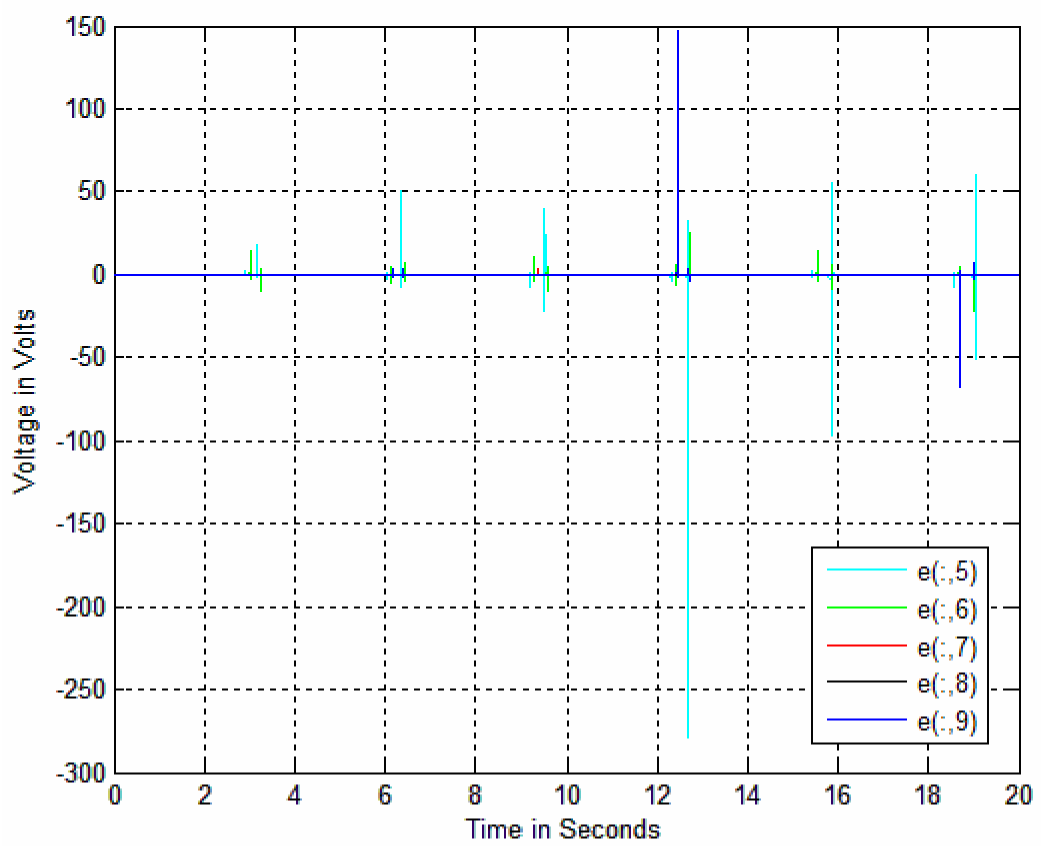

Figure 8- Relative error between X_r and X_{opt}

Clearly the errors overlap at some of the time intervals. The spikes seen in this graph are due to the fact that we are plotting the relative error and dividing by $X_r(i, j)$ which, at some points in time, when the voltages across the $N+1=5$ parasitic capacitors are approaching zero, have very small values that are reflected at spikes in the graph.

For example, the minimum of $err(:,5)$ is -278.6434, this minimum occurs at index $i = 6346$ which corresponds to $T(6346) = 12.69$ seconds. Hence we have $err(6346,5) = -278.6434$, which corresponds to

$err(6346,5) = \dfrac{X_r(6346,5) - X_{opt}(6346,5)}{X_r(6346,5)}$, where $X_r(6346,5) = \text{-}1.2786\text{e-}007$ V and

$X_{opt}(6346{,}5) = \text{-}3.5755\text{e} - 005\ \text{V}$. We can see that the difference between these two values is very small and both of them are very close to zero however $X_r(6346{,}5)$ is of order 10^{-7} and $X_{opt}(6346{,}5)$ is of order 10^{-5} which introduces the spike of -278.6434 in the relative error plot.

Plotting the average of the error between the two set of states X_r and X_{opt}, we obtain the following:

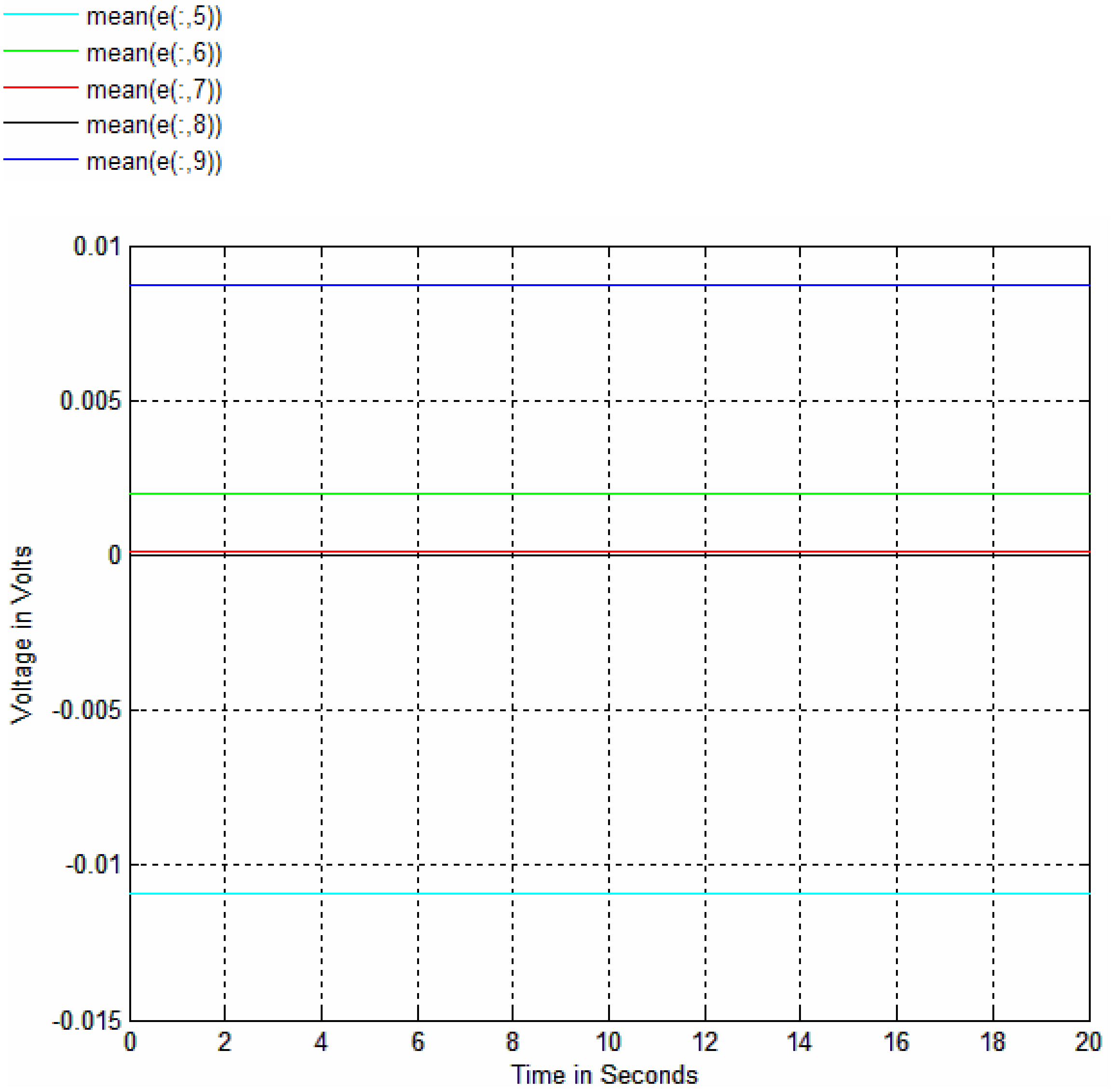

Figure 9- Average error plot between X_r and X_{opt}

As we can see in the figure, the averages are small having absolute values smaller than 1.1%, more precisely:

$$\left|mean(err(:,5))\right| = 1.09\%,\ \left|mean(err(:,6))\right| = 0.2\%,\ \left|mean(err(:,7))\right| = 0.0112\%,$$

$$\left|mean(err(:,8))\right| = 4.8104\text{e}-4\%,\ \left|mean(err(:,9))\right| = 0.8729\%.$$

These results show that the parasitic capacitor values obtained using the least-squares nonlinear optimization algorithm yield a set of state trajectories that closely follow our reference model X_r with a relatively negligible error.

3.5 Conclusions

We have seen in this chapter that using the least-squares nonlinear optimization algorithm yields an optimal set of parasitic capacitor values that, when used for input to the Marx generator, result in a state model that closely follows the reference model with a relatively negligible error.

Chapter 4 Generating a new reference model

We introduce in this chapter some of the properties associated with the Marx generator state trajectories and develop a state shifting algorithm that shifts the state trajectories by a specific shifting factor such that the spark generated by the generator occurs at a user specified time instant.

4.1 Generating a new reference state model:

It is worth noting that the trajectories of the 14 states have all the following properties:

1. They are periodic with period T_{per}.
2. Symmetric with respect to, depending on the state in consideration, either their minimum or maximum value which occurs at $t_f = \frac{T_{per}}{2}$ seconds for each T_{per} time interval.

As an illustration of these two properties, consider the voltage across the $N+1$ capacitor, which corresponds to the 9th state of our model:

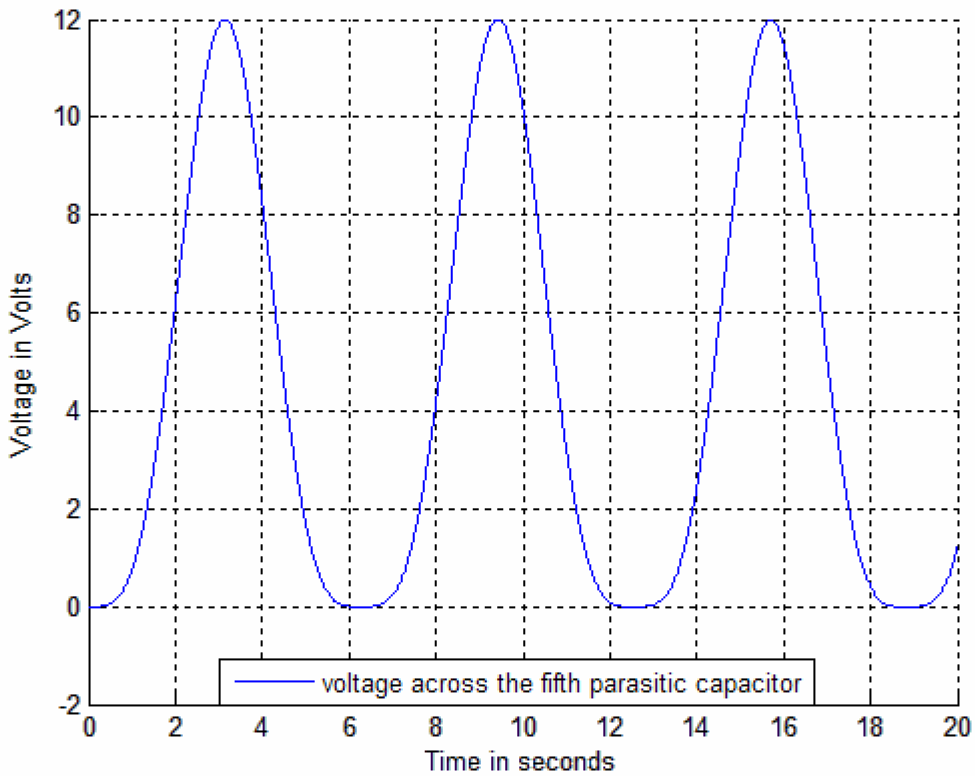

Figure 10- Voltage across the 5th parasitic capacitor

Clearly the state trajectory is periodic with period $T_{per} = 6.284$ seconds.

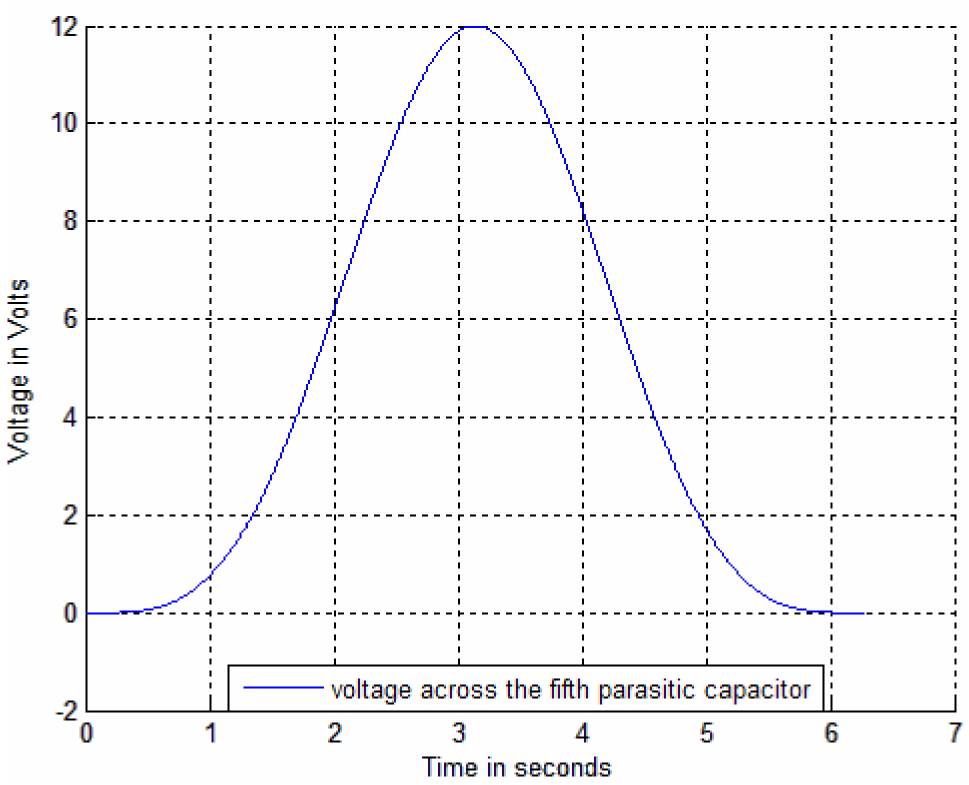

Figure 11- Voltage across the 5th parasitic capacitor up to T_{per}

The maximum during the first period is 11.9999 at $t_f = \frac{T_{per}}{2} = 3.142$ seconds,

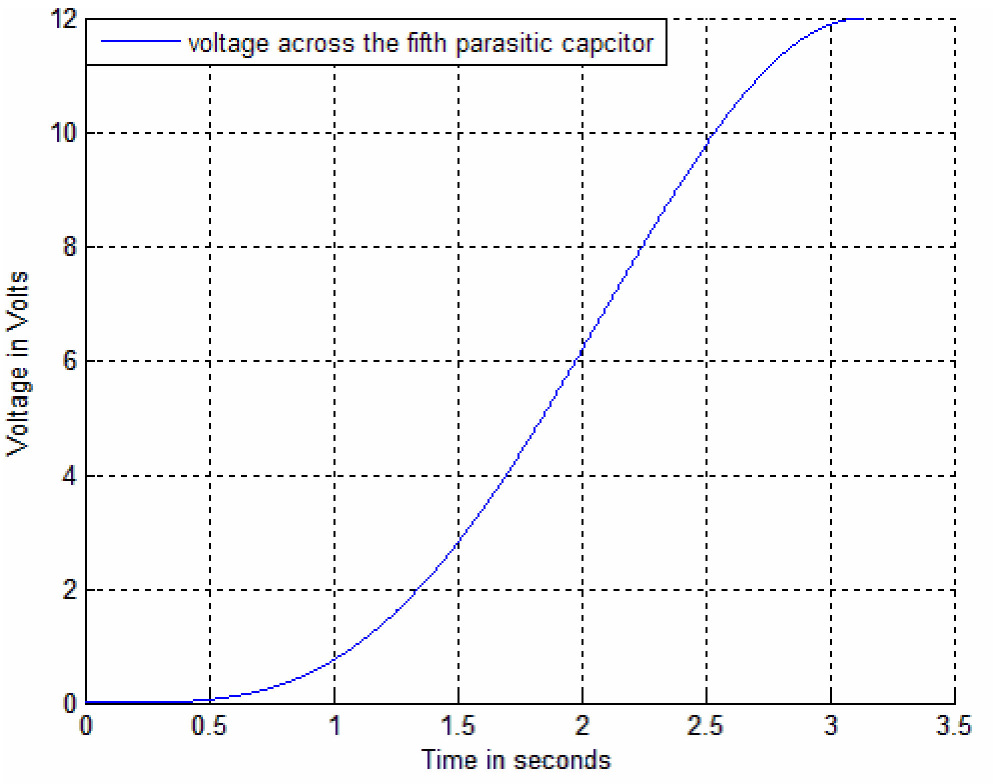

Figure 12- Voltage across the 5th parasitic capacitor up to t_f

Having this portion of the state up to $t_f = 3.142$ seconds, we can rebuild the state trajectory for the first period using the symmetrical property with respect to the time index at which the maximum occurs.

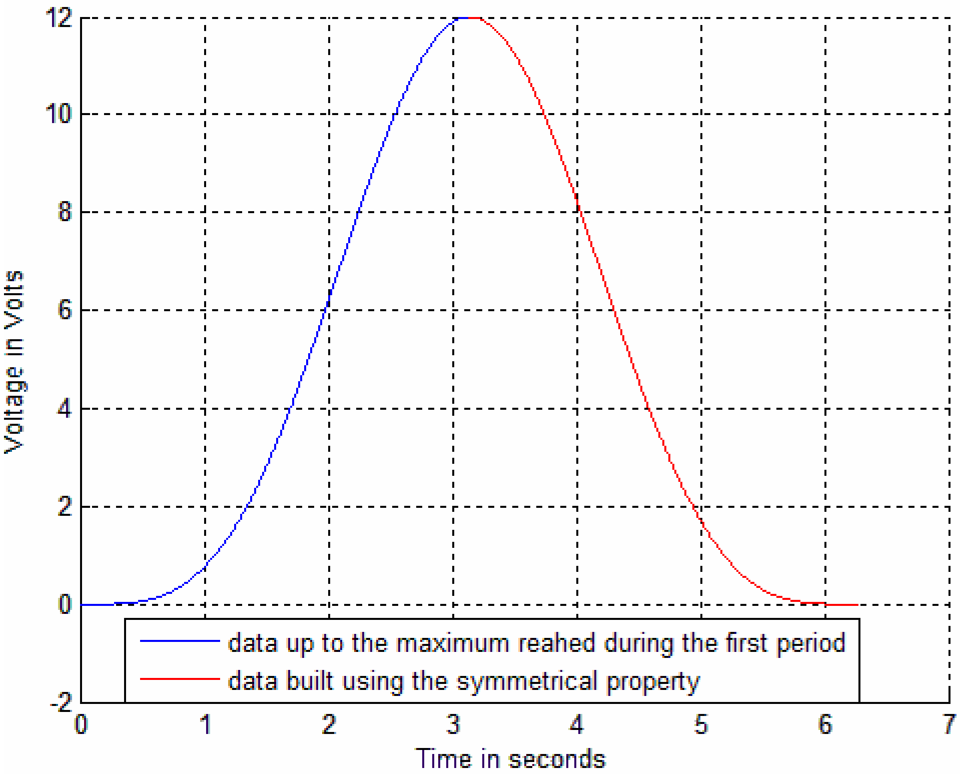

Figure 13- Rebuilding $Vc_5^{'}$ up to T_{per} using symmetry with respect to t_f

Having rebuilt the data up to the first period, the full state trajectory can be generated by adding up the rebuilt data for successive multiples of the period up to $T = 20$ seconds:

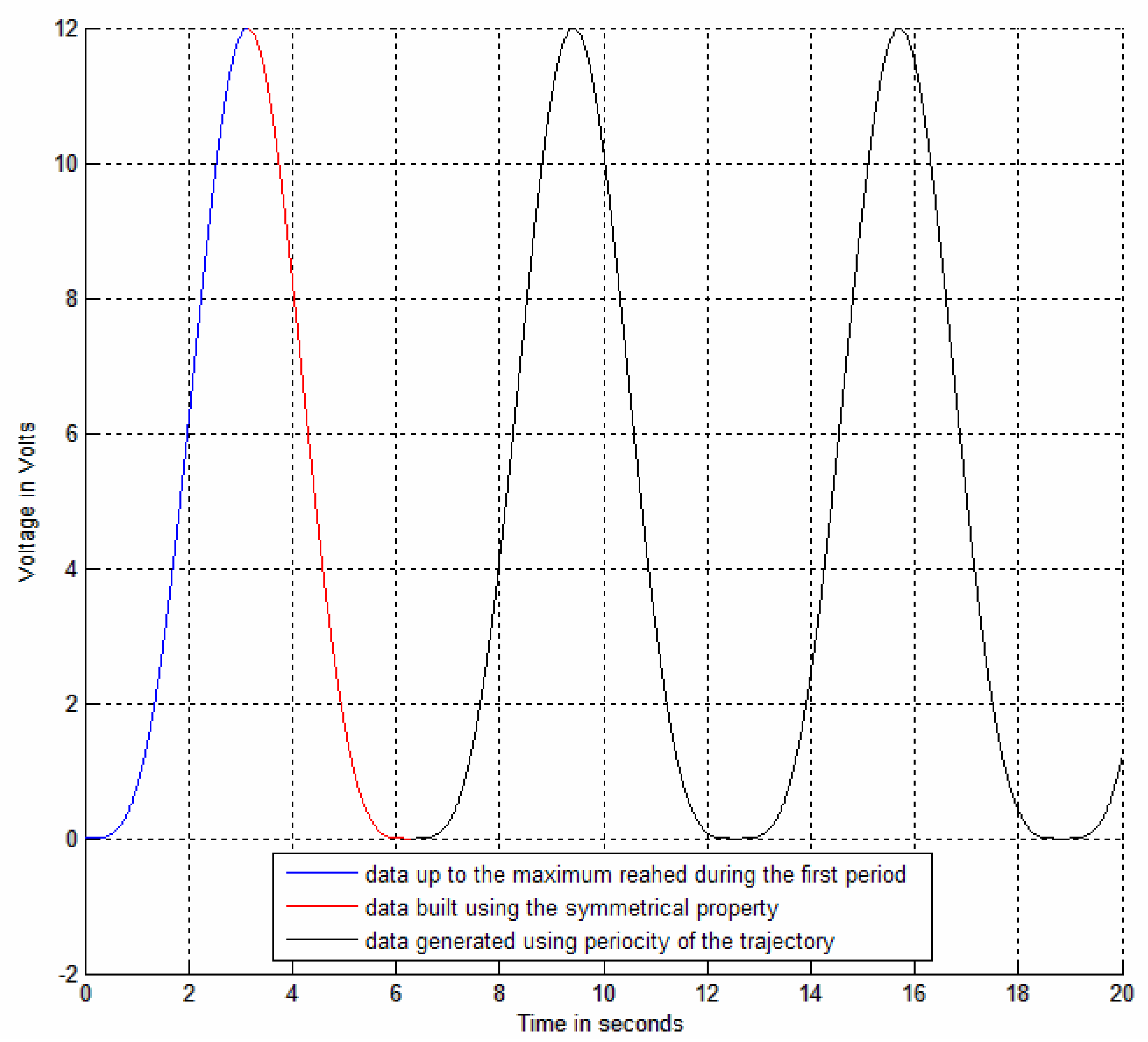

Figure 14- Rebuilding $Vc_5^{'}$ up to $T_{total} = 20$ seconds using the periodicity property

Having shown the symmetrical and periodical properties of one of the states, in particular the voltage across the $N+1$ capacitor, we will display one more example of these properties at the 14th state which represents the current across the $N+1$ inductor. The corresponding reference trajectory is displayed in the following figure:

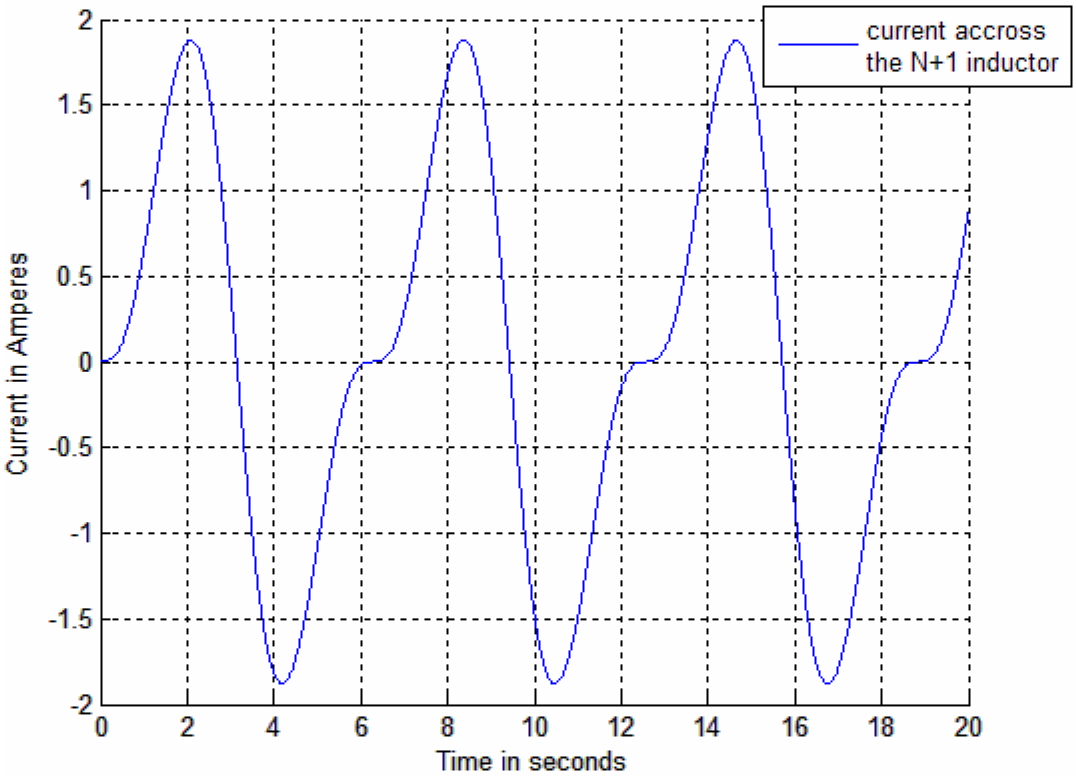

Figure 15- Current across the fifth inductor

Again by looking at Figure 15 we can see that the trajectory is periodic having the same period as the 9th state, that is $T_{per} = 6.284$ seconds. Displaying the trajectory for the first period:

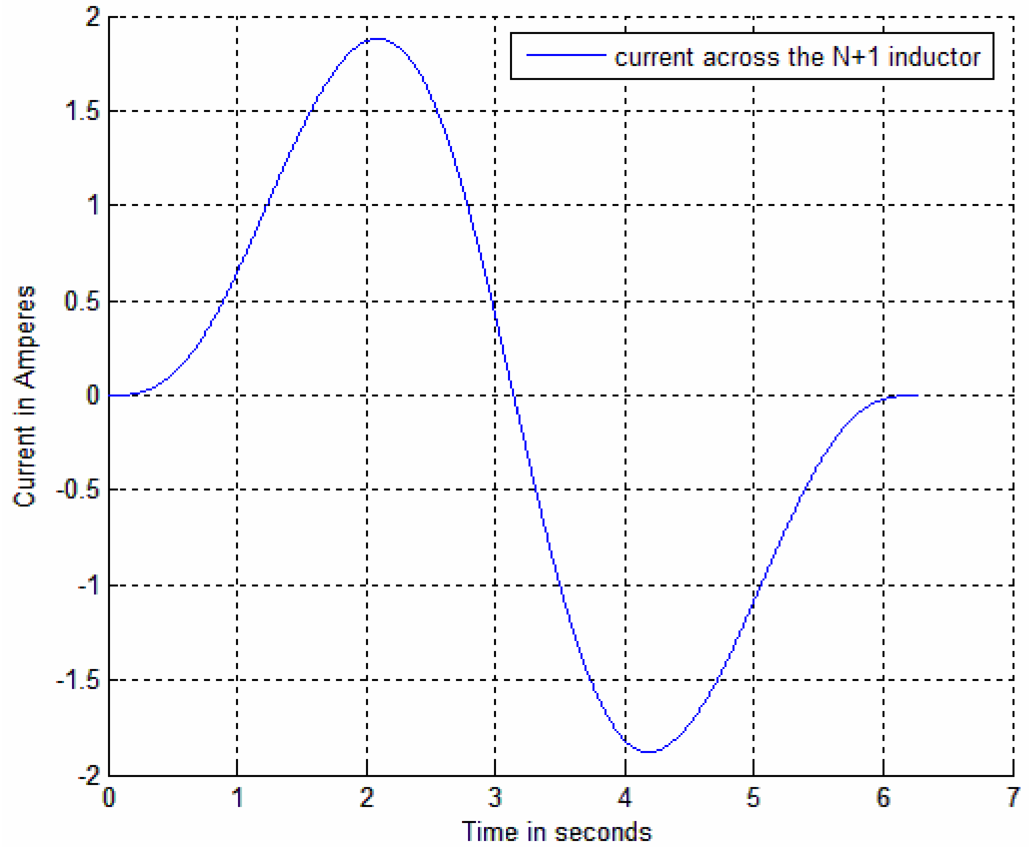

Figure 16- Current across the fifth inductor up to T_{per}

This trajectory is symmetric with respect to its minimum at $t_f = \frac{T_{per}}{2} = 3.142$ seconds.

Plotting the curve to $t_f = 3.142$ seconds:

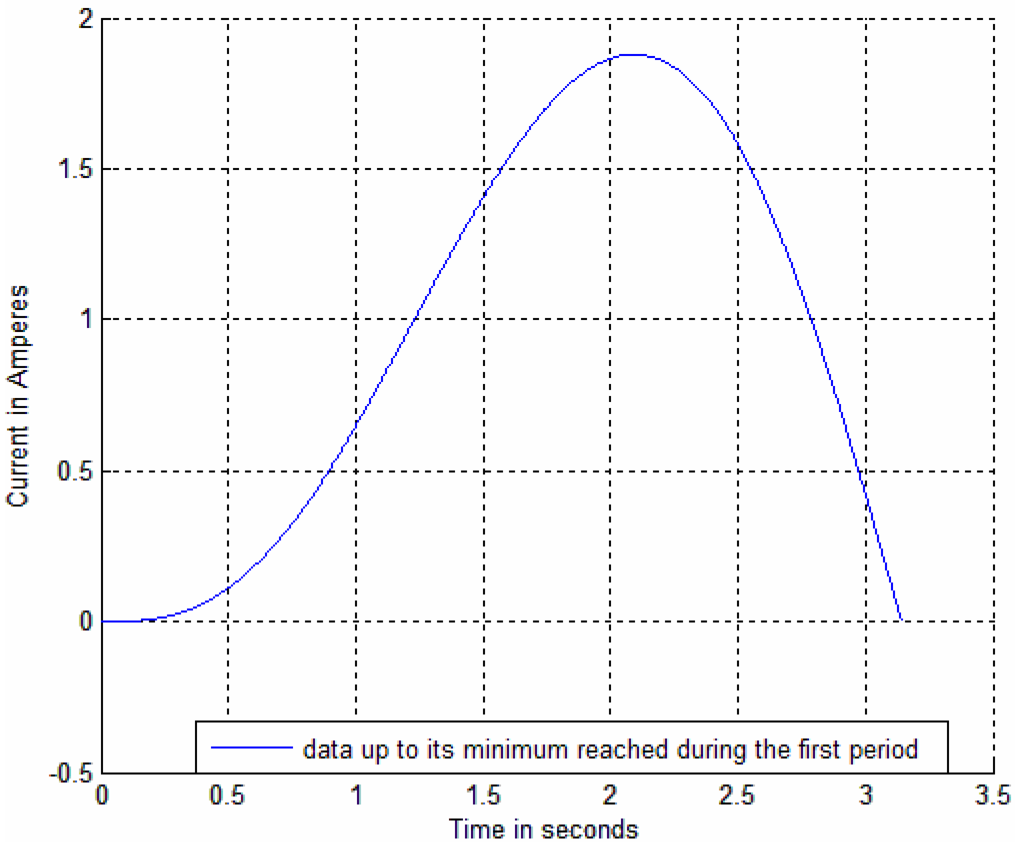

Figure 17- Current across the fifth inductor up to t_f

Rebuilding the state path up to the first period T_{per} can be achieved by using the symmetrical property of the curve with respect to the time index at which the minimum occurs:

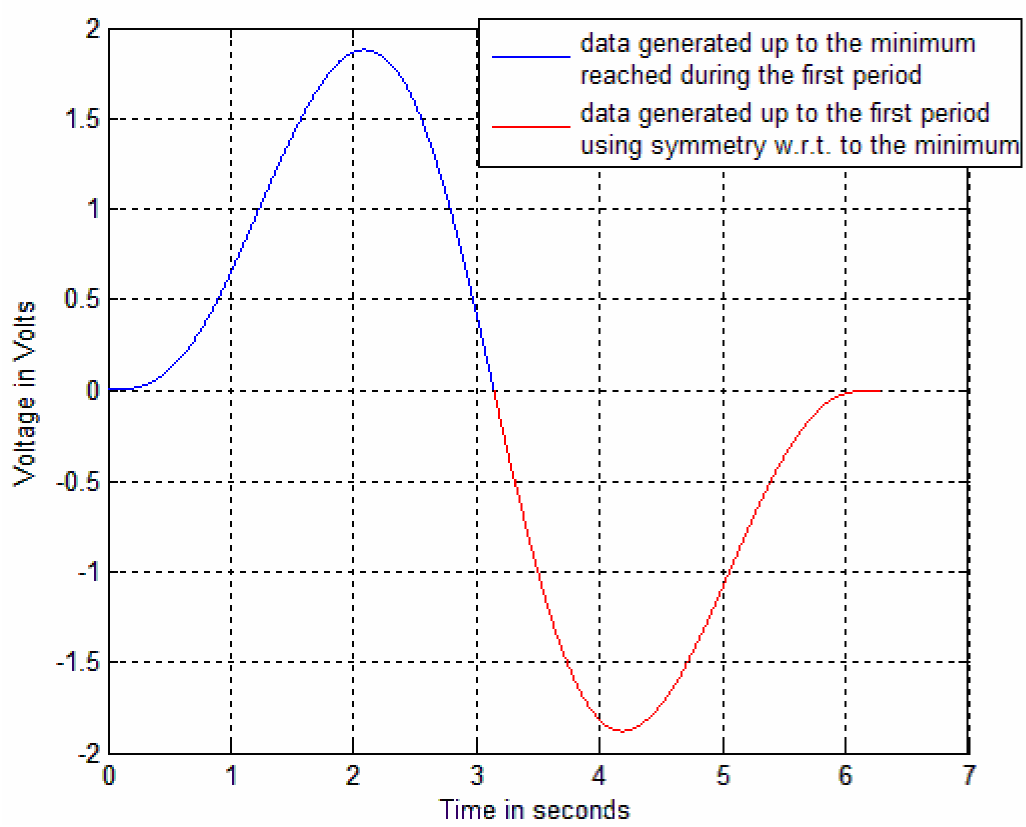

Figure 18- Rebuilding $Ic_5^{'}$ up to T_{per} using symmetry with respect to t_f

The full path can be, hence, built using the periodicity of the original curve as is explained for the 9th state trajectory:

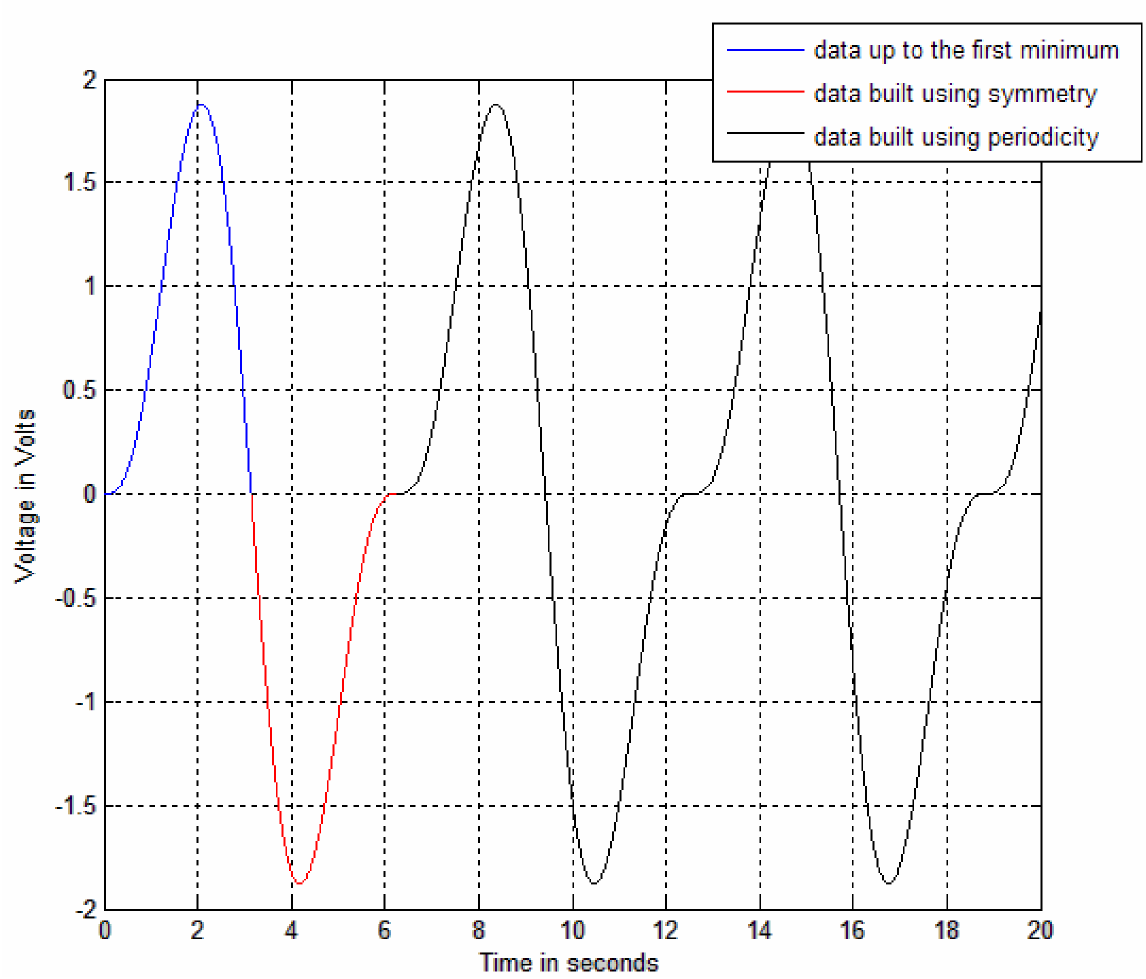

Figure 19- Rebuilding $Ic_5^{'}$ up to $T_{total} = 20$ seconds using the periodicity property

4.2 Generating the New Reference Model

The next step to be considered is generating a new X_n; that is a desired trajectory to be followed, during which the peak voltage at the $N+1=5^{th}$ capacitor is set to be at a new desired time value. Hence, this would require shifting all of the 14 states of the $N=4$ stages Marx generator by the right fraction of time. As an example, knowing that in our reference model $t_f(i=1572)=3.14$ seconds, and wanting the maximum of the 9th state during the first period to occur at $T(i=1402)=2.802$ seconds, than the factor by which the state trajectories should be shifted by is $fc = \frac{1402}{1572}$. To this extent and exploiting the stated state trajectory properties of symmetry and periodicity, we developed the following two steps algorithm to reflect the desired state shift:

1st step:

- Define both the maximum and minimum during the first period and their respective indices in a variable called temp.
- Define a new array called Xnd_temp in which values ranging from 0.01% up to 100% of the maximum corresponding state variable are stored.
- Go through the entries in Xnd_temp and compare them to actual values in X_d in order to determine their respective indices in X_d. At this stage we consider two scenarios
    - The value in Xnd_temp is exactly equal to one of the entries of X_d in this case its entry index would be the same as in X_d.
    - The value in Xnd_temp does not have an exact match in X_d. Therefore it will eventually be in an interval between a lower and higher bound values in X_d. Hence its stored entry index is chosen to be equal to the lower bound entry index of its interval.

The 2nd step:

- We define the shift factor fc.
- We multiply the corresponding maximum value index by fc, round the result and store it in $new\_\max\_i$.

- Similarly, we multiply the corresponding minimum value index by fc, round the result and store it in $new\_\min\_i$.
- Store at $X_n(new\_\max x\_i, j)$ the corresponding maximum value of X_d.
- Store at $X_n(new\_\min\_i, j)$ the corresponding minimum value of X_d.
- Move the values stored in Xnd_temp into X_n with the corresponding indices being multiplied by fc.
- Until now we have built X_n for half a period, therefore we use the symmetric property of X_d to built X_n for a one period duration such that the new period is $T_{per} = \left(new\_\max\_i\ (or\ new\_\min\_i)\right)\times 2$.
- Using the periodicity property of X_d, we fill out the remaining entries of X_n up to $T_{total}(i = 10001) = 20$ seconds.

4.3 Conclusion:

This chapter showed the symmetrical and periodical properties of the $2N+1$ states and presented the algorithm used to generate a new reference trajectory X_n based on these properties and the reference model X_r, such that the maximum of the $N+1$ parasitic capacitor occurs at a desired point in time.

Chapter 5 Implementation and Results

In this chapter, we apply the state shifting algorithm developed in Chapter 4 to the reference state model of the $N = 4$ stages Marx generator using two different shifting factors. Hence, we obtain two new reference state trajectory models with different voltage peaking times at the $N + 1 = 5^{th}$ parasitic capacitor. Next, we apply the least-squares nonlinear optimization algorithm explained in 23Chapter 3 to obtain the set of parasitic capacitor values that best track the corresponding reference model and present a comparison between the state trajectory obtained using the optimization results and the corresponding reference model.

5.1 Case 1: Maximum voltage across fifth capacitor at T= 2.802 seconds

Using the previous algorithm, we first generated a new X_{n1} where the maximum voltage across the $N + 1 = 5^{th}$ parasitic capacitor occurs at $T(i = 1402) = 2.802$ seconds, i.e.

$$fc = \frac{1402}{1572} \text{ and } X_{n1}(1402,9) = 12V.$$

By using the algorithm developed in the previous chapter the new state trajectory X_{n1} looks as follows:

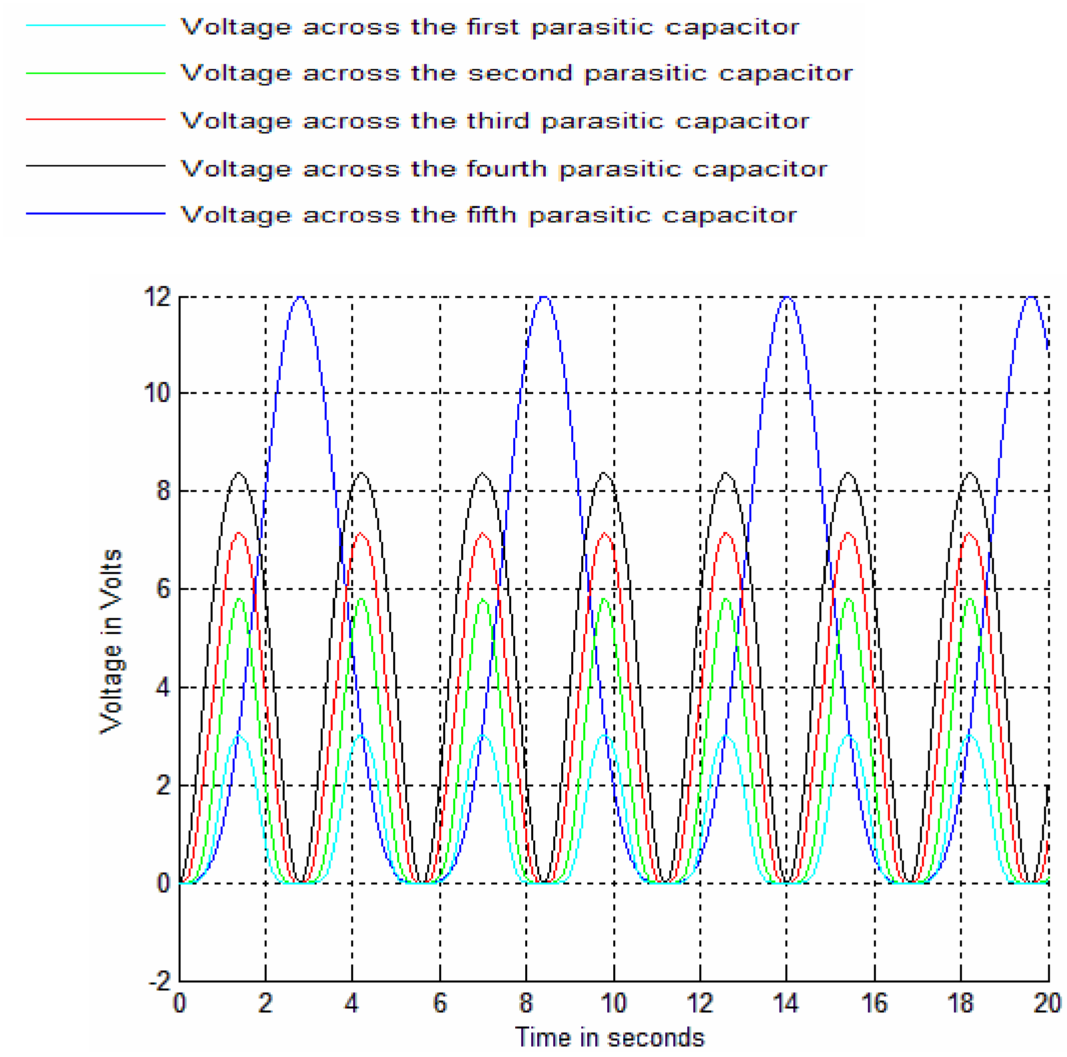

Figure 20- New state model X_{n1}

The subsequent figure reflects better the time shift applied to the new state trajectory. To avoid confusion, I will only plot the voltage across the 9^{th} parasitic capacitor in X_r and X_{n1}:

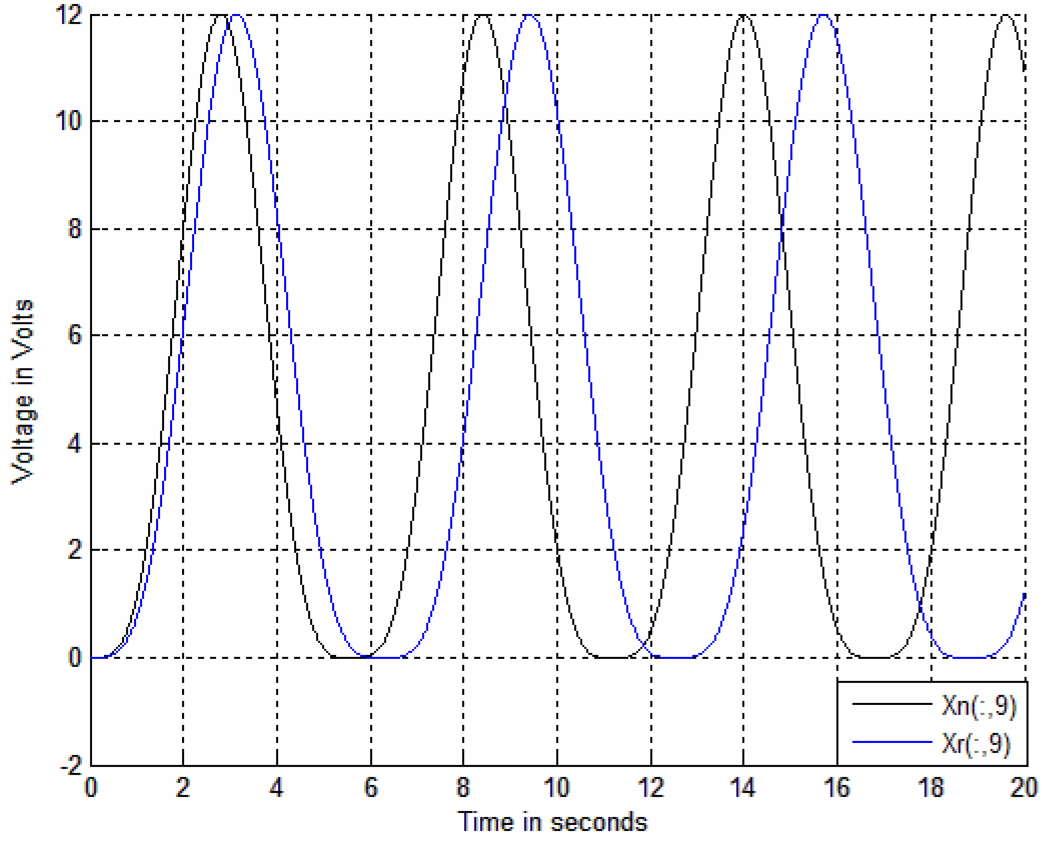

Figure 21- Voltage across the fifth parasitic capacitor in X_{n1} lagging the voltage across the fifth parasitic capacitor in X_r

Please note that this same time shift of $fc = \frac{1402}{1572}$ is reflected in all of the 14 states of our $N = 4$ stages Marx generator. You can see from the figure that the maximum of $X_{n1}(:,9)$ is leading the maximum of our reference model $X_r(:,9)$ by $T(1572) - T(1402) = 3.142 - 2.802 = 0.340$ seconds.

Using the nonlinear least-squares optimization algorithm, the optimal set of parasitic capacitor values that best follows the new state trajectories in X_{n1} is the following:

$$C_1^{'} = 0.032176\,\mathrm{F.},\ C_2^{'} = 0.060065\,\mathrm{F.},\ C_3^{'} = 0.02111\,\mathrm{F.},\ C_4^{'} = 0.1115\,\mathrm{F.}$$

Using these values as inputs for the parasitic capacitor values and multiplying both C and L by fc to reflect the time shift into the resonant frequency ω_0, we obtain the following state trajectories stored in X_{opt} :

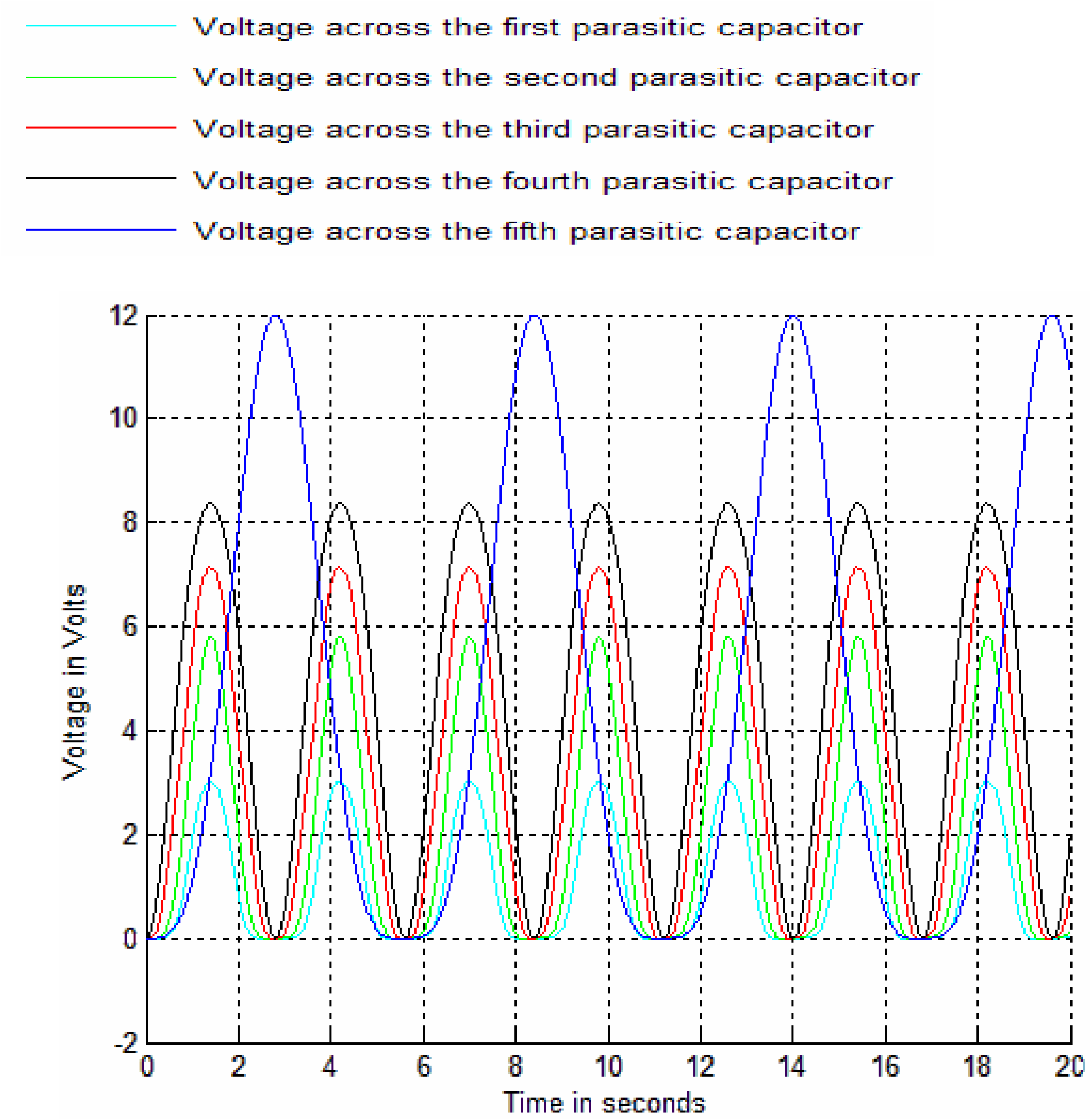

Figure 22- State trajectories X_{opt} using the optimal set of parasitic capacitors

The states generated using the new set of parasitic capacitors seems to be closely following the trajectories in X_{n1} shown in Figure 20. The relative error between he two set of state trajectories X_{n1} and X_{opt} is reflected in the following plots:

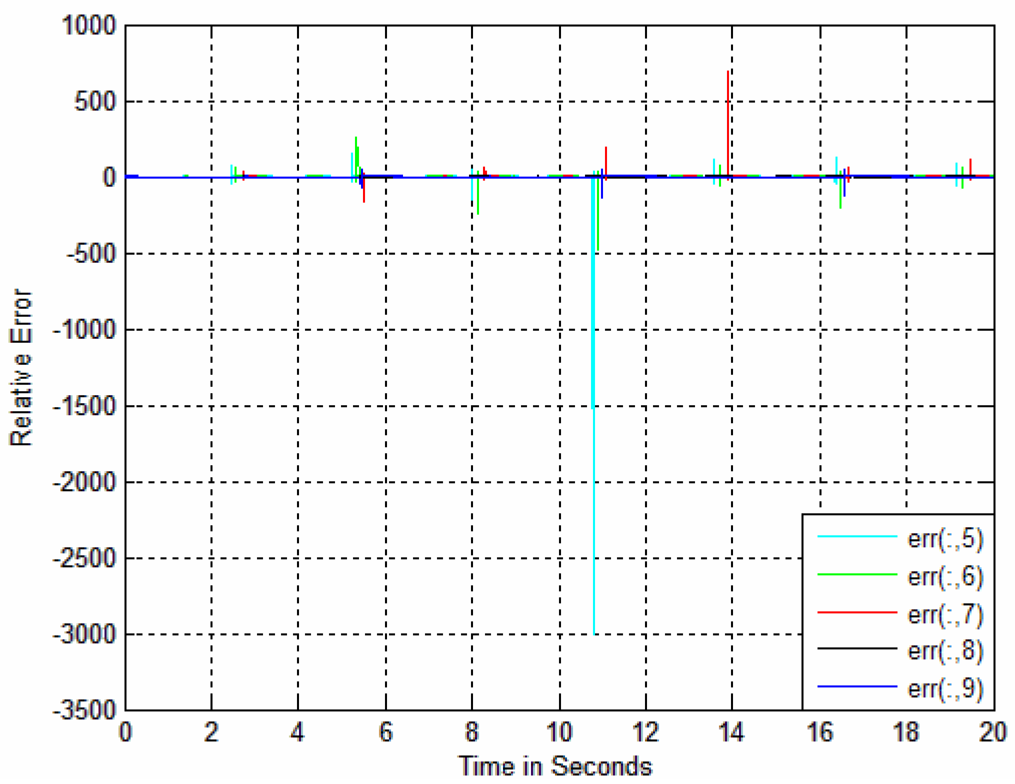

Figure 23- Relative error between X_{n1} and X_{opt}

As is reflected in the figure, the spikes of the errors along the different states have larger magnitudes at certain points in time than the relative error plot shown in Figure 8. The absolute maximum magnitudes reached by the error between the, new, reference state trajectories X_{n1} and the simulated ones X_{opt} are as follows:

$$\max|(err(:,5))| = 3005.8,\ \max|(err(:,6))| = 479.2271,\ \max|(err(:,7))| = 684.7525,$$

$$\max|(err(:,8))| = 1,\ \max|(err(:,9))| = 142.0193.$$

The next plot shows the mean of the relative error between the states in X_{n1} generated using the algorithm in the previous chapter and the ones obtained by simulation using the optimal set of parasitic capacitors and stored in X_{opt} :

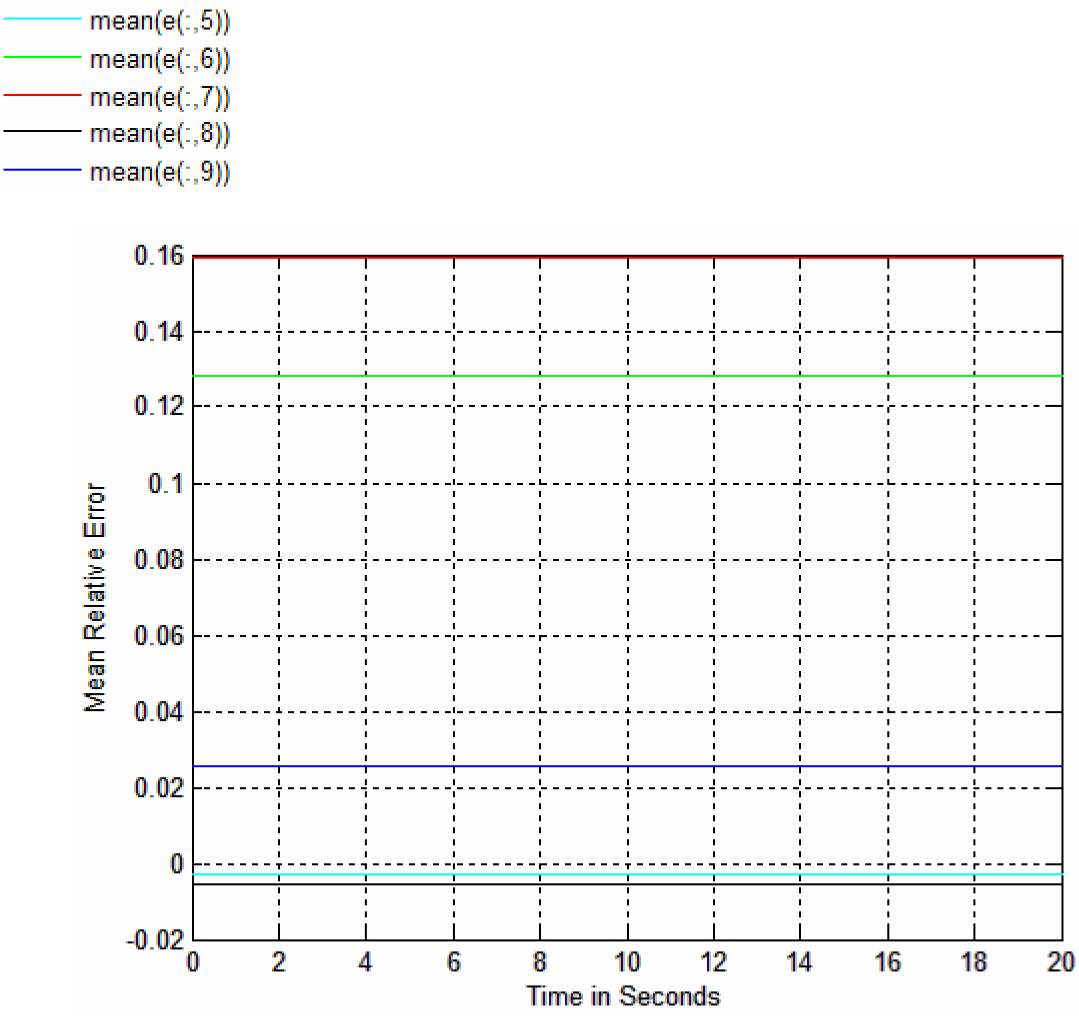

Figure 24- Average error between X_{n1} and X_{opt}

More precisely the absolute values of the average percentages are:

$$\left|mean(err(:,5))\right| = 0.2753\%,\ \left|mean(err(:,6))\right| = 12.8018\%,\ \left|mean(err(:,7))\right| = 15.9186\%,$$

$$\left|mean(err(:,8))\right| = 0.5530\%,\ \left|mean(err(:,9))\right| = 2.5711\%.$$

The above percent averages are small and relatively negligible except for the 6th and 7th states that have a bit higher average percent error.

If we consider the 5th state more closely, we have seen that it has a very large peek of 3005.8, however it has an overall average relative error of 0.2753% which shows that the peek happens at a fixed instant of time, i.e. at $T(i = 5393) = 10.7840$ whereas for the remaining time indices the corresponding relative error is very small.

It is worth noting that the spikes in the relative error magnitude occur at points in time where the voltages across the $N + 1 = 5$ parasitic capacitors approach zero volts. Hence, choosing smaller time intervals during which there are no sudden jumps in the error magnitude decreases the mean error between any two pair of states significantly.

To better illustrate this statement, the next figure, simultaneously, displays the voltage across the first parasitic capacitor (i.e. the 5th state in X_{opt}) and the relative error between $X_{n1}(:,5)$ and $X_{opt}(:,5)$:

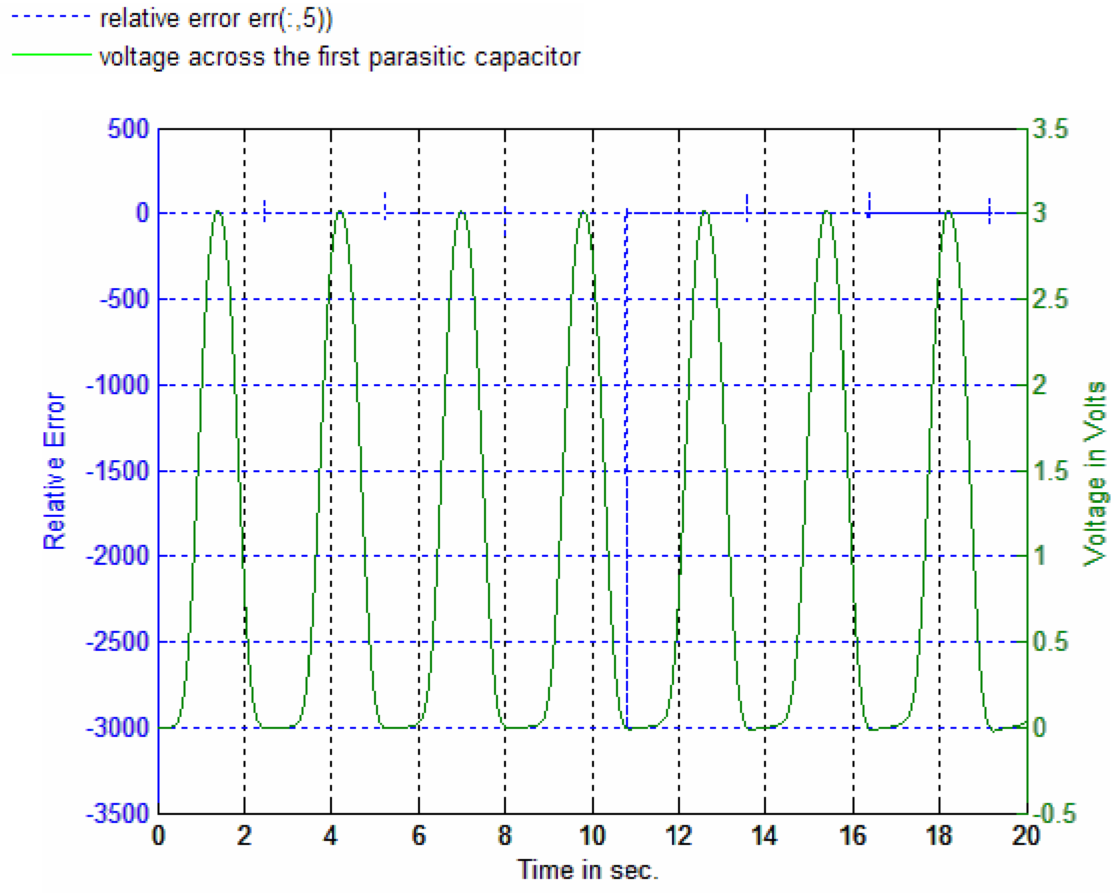

Figure 25- Voltage across the first capacitor and the corresponding relative error

As stated the spikes occur when the state approaches zero volts. Choosing a time interval during which the voltage across the fifth state in X_{opt} is not approaching zero, in particular between 15 and 16 seconds and repeating the same plot as above:

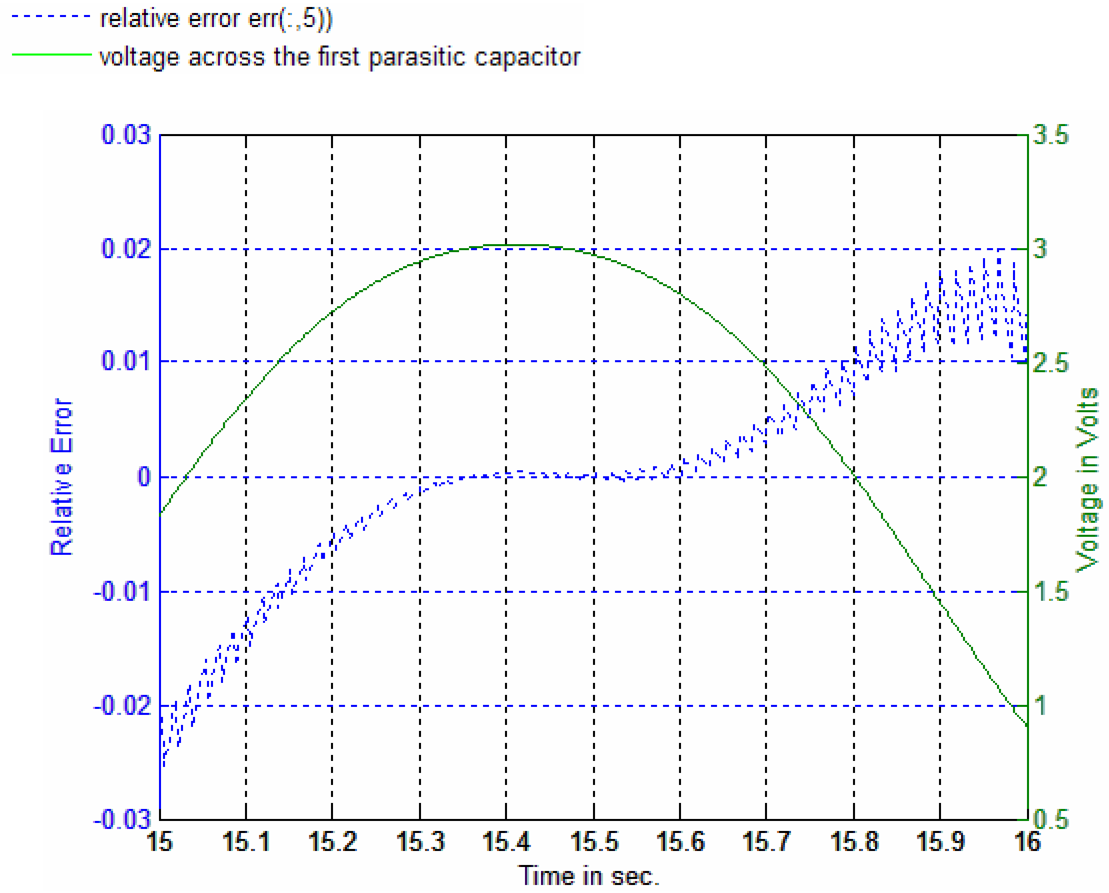

Figure 26- Voltage across the first capacitor and the corresponding relative error between for $15\text{sec.} \le T \le 16\text{sec.}$

As seen in this figure, the relative error has a small magnitude in this interval:

$$-0.0253 \le err(7501:8001,5) \le 0.0198$$

and absolute mean percentage of:

$$\left| mean(err(7501:8001,5)) \right| = 0.0426\%$$

Where $T(7501) = 15$ seconds and $T(8001) = 16$ seconds.

The spikes observed in Figure 23 are due to numerical discrepancies induced by the relative error calculation. For instance, if we again consider in more details the maximum of the relative error corresponding to the 5th state:

$$\max\left|(err(:,5))\right| = 3005.8$$

This occurs at index 5393 of the err matrix, i.e. $err(5393,5) = 3005.8$. Knowing that

$$err(i,j) = \frac{X_{n1}(i,j) - X_{opt}(i,j)}{X_{n1}(i,j)}, \text{ for } 1 \le i \le 10001 \text{ and } 1 \le i \le 14$$

Hence, $err(5393,5) = \dfrac{X_{n1}(5393,5) - X_{opt}(5393,5)}{X_{n1}(5393,5)}$, where

$X_{n1}(5393,5) = 5.8184\text{e-}006$ and $X_{opt}(5393,5) = 1.75 \times 10^{-2}$. Both of these values have small magnitudes, however one of them has order 10^{-2} and the other 10^{-6} which induced the large jump in the relative error magnitude. To overcome this issue, I will set a threshold value such that whenever the order of X_{n1} is less than 10^{-3} while the order of X_{opt} is less than 10^{-2}, set the corresponding relative error value to zero. Hence, with the application of this constraint, the absolute maximums of the relative state errors become:

$$\max\left|(err(:,5))\right| = 19.3360,\ \max\left|(err(:,6))\right| = 1,\ \max\left|(err(:,7))\right| = 0.9865,$$

$$\max\left|(err(:,8))\right| = 0.3239,\ \max\left|(err(:,9))\right| = 0.9475.$$

with the following absolute mean percentages:

$$mean\left|(err(:,5))\right| = 2.1227\%,\ mean\left|(err(:,6))\right| = 3.1654\%,\ mean\left|(err(:,7))\right| = 0.8188\%,$$

$$mean\left|(err(:,8))\right| = 0.4509\%,\ mean\left|(err(:,9))\right| = 1.3447\%.$$

Thus the absolute maximum and percentage average of the relative errors have been greatly decreased, which shows the effect of calculation discrepancies on the error especially in regions where the state variable values become minute when approaching zero.

5.2 Case 2: Maximum voltage across fifth capacitor at T= 3.302 seconds

The next case considered is when the maximum voltage across the $N+1=9^{th}$ parasitic capacitor occurs at $T(i=1652)=3.302$ seconds, i.e.

$$fc=\frac{1652}{1572} \text{ and } X_{n2}(1652,9)=12V.$$

Hence, the new state trajectories look as follow:

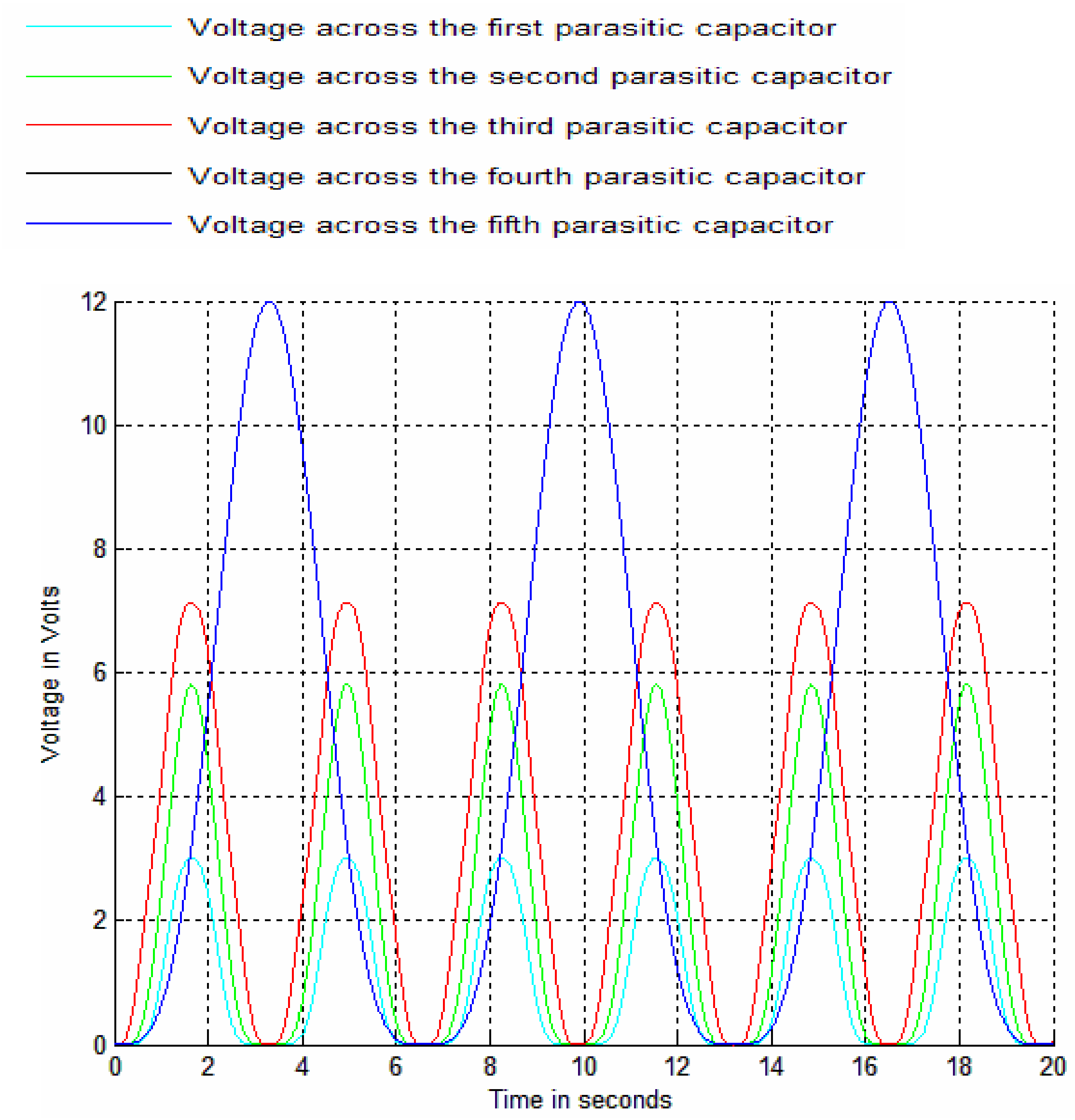

Figure 27- New state trajectories X_{n2}

The subsequent figure reflects better the time shift applied to the new state trajectory. To avoid confusion, I will only plot the voltage across the 9^{th} parasitic capacitor in X_r and X_{n2}:

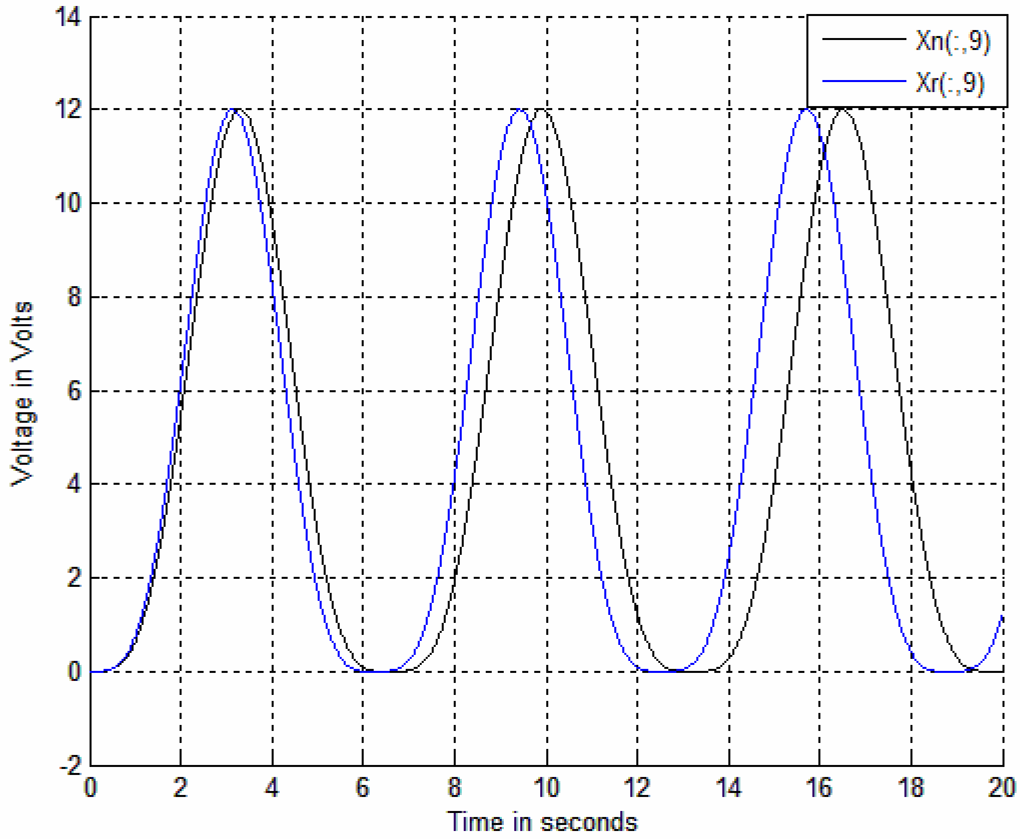

Figure 28- Voltage across the fifth parasitic capacitor in X_{n2} leading the voltage across the fifth parasitic capacitor in X_r

Please note that this same time shift of $fc = \frac{1652}{1572}$ is also reflected in all of the 14 states of our N=4 stages Marx generator. You can see from the figure that the maximum of $X_{n2}(:,9)$ is lagging the maximum of our reference model $X_r(:,9)$ by $T(1652) - T(1572) = 3.302 - 3.142 = 0.16$ seconds.

Again using the nonlinear least-squares optimization algorithm, the optimal set of parasitic capacitor values that best follows the new state trajectories X_{n2} is the following:

$$C_1^{'} = 0.037819\,\text{F.},\; C_2^{'} = 0.070636\,\text{F.},\; C_3^{'} = 0.024877\,\text{F.},\; C_4^{'} = 0.13122\,\text{F.}$$

Using these values as inputs for the parasitic capacitor values and multiplying both C and L by fc to reflect the time shift into the resonant frequency ω_0, we obtain the following state trajectories stored in X_{opt}:

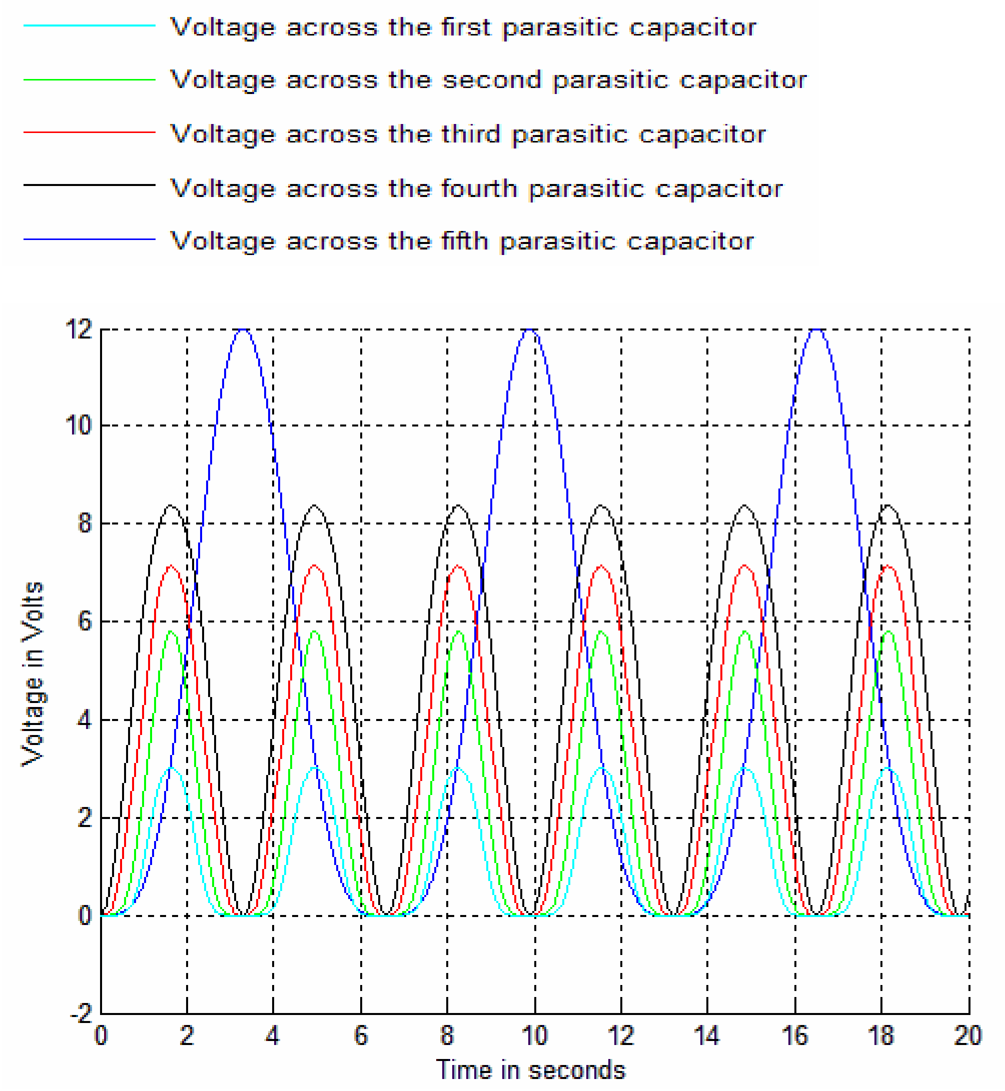

Figure 29- State trajectory X_{opt} obtained using the set of optimal parasitic capacitors

From the above plot, the state trajectories seam to follow the new reference model X_{n2} much better than the previous case. The resulting relative error plots are as follows:

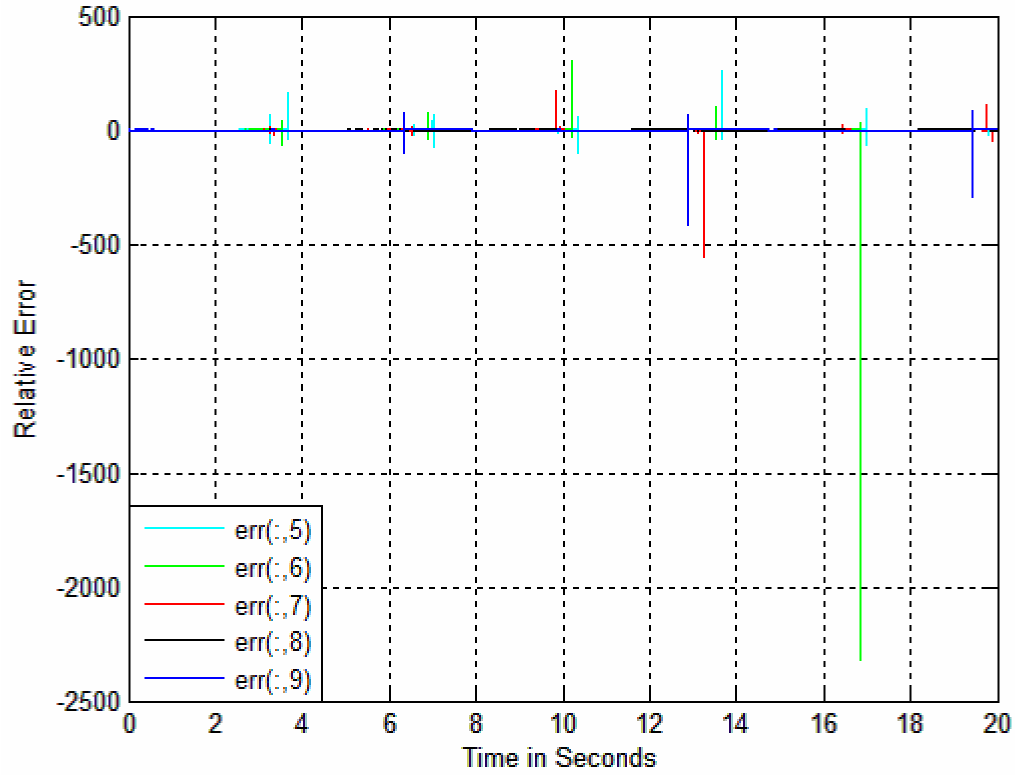

Figure 30- Relative error between X_{n2} and X_{opt}

Still one can notice the jumps in the magnitude of the relative error that are larger than the ones shown in Figure 8, however much smaller than the spikes obtained in the previous study case. The absolute maximums of the relative error magnitudes are as follows:

$$\max|(err(:,5))| = 258.8794,\ \max|(err(:,6))| = 2319.9,\ \max|(err(:,7))| = 558.8778,$$

$$\max|(err(:,8))| = 1,\ \max|(err(:,9))| = 418.3080.$$

The subsequent plot shows the mean of the relative error between the states in X_{n2} and the ones obtained by simulation X_{opt}:

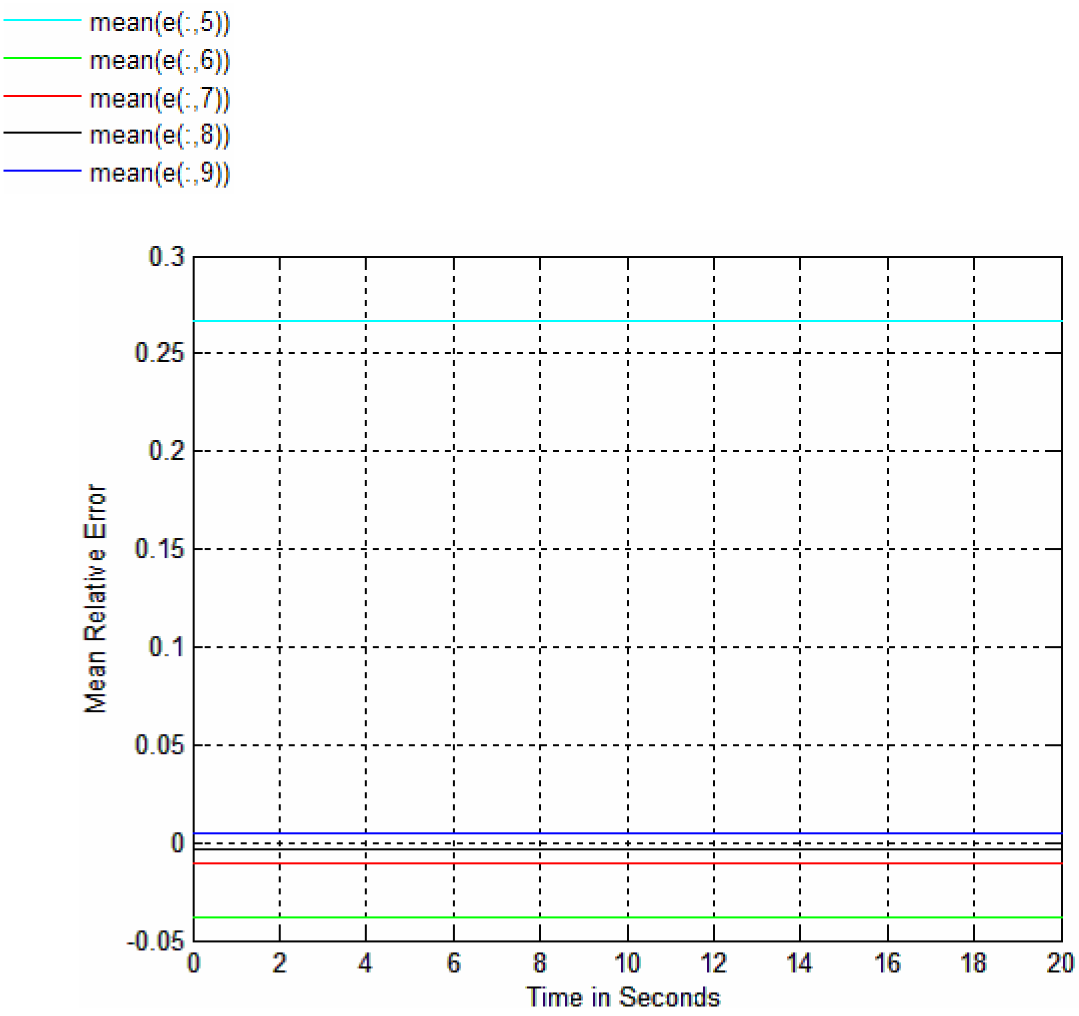

Figure 31- Average error between X_{n2} and X_{opt}

The absolute values of the averages are:

$$|mean(err(:,5))| = 26.6565\%,\ |mean(err(:,6))| = 3.8268\%,\ |mean(err(:,7))| = 1.0534\%,$$

$$|mean(err(:,8))| = 0.3932\%,\ |mean(err(:,9))| = 0.5068\%.$$

Similarly to the reasoning behind the first case study, the percentages shown above are clearly small enough for the exception of the one corresponding to the 5th state. The large average error value is due to the larger spikes in the corresponding relative error as shown in Figure 30.

Again the spikes in the relative error magnitude occur at points in time where the voltage across the $N+1$ parasitic capacitors approaches zero volts.

Similarly to the previous case, choosing smaller time intervals during which there are no sudden jumps in the error magnitude decreases the mean error between any two pair of states significantly. The following figure, gives a better pictorial explanation of this fact by simultaneously displaying the voltage across the fifth parasitic capacitor (i.e. the 9th state in X_{opt}) and the relative error between $X_{n2}(:,9)$ and $X_{opt}(:,9)$:

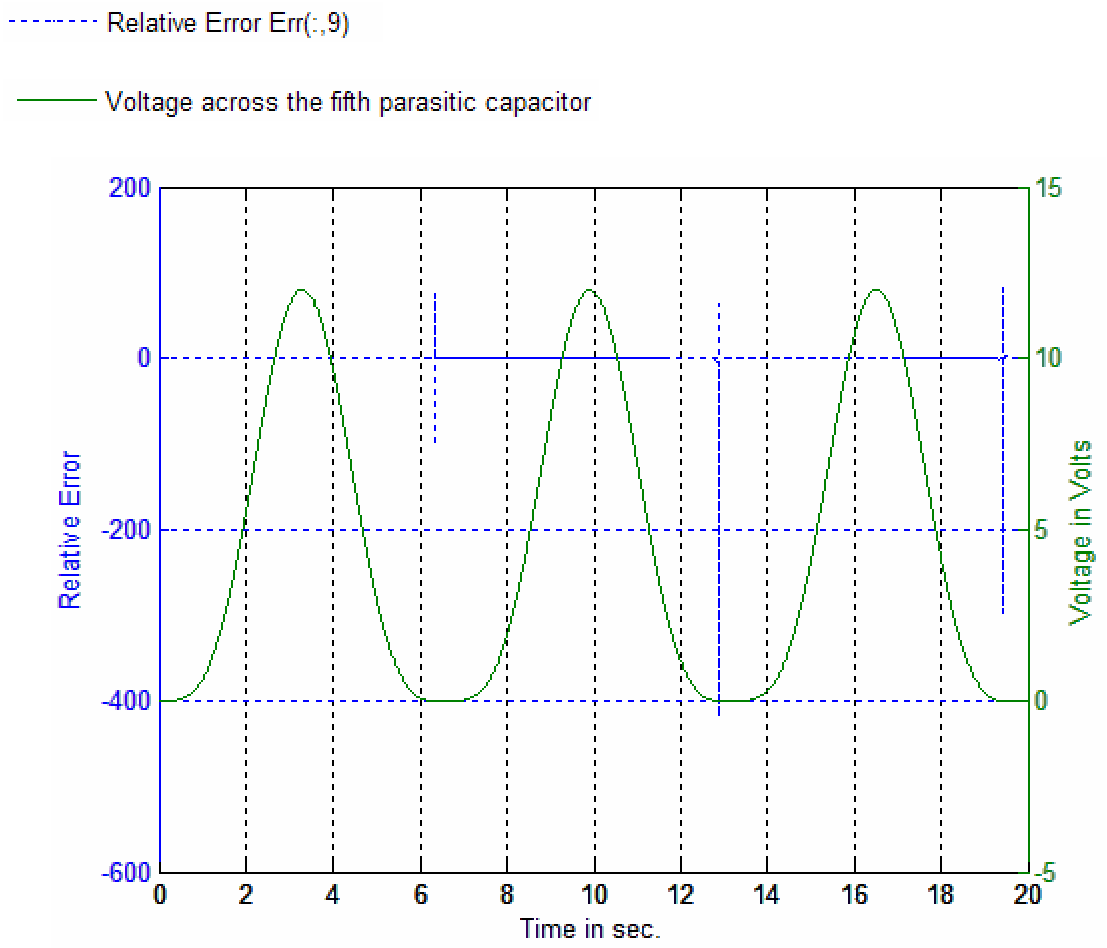

Figure 32- Voltage across the ninth capacitor and the corresponding relative error between

As stated the spikes occur when the state approaches zero volts. Choosing a time interval during which the voltage across the ninth state in X_{opt} is not approaching zero, in particular between 9 and 12 seconds and repeating the same plot as above:

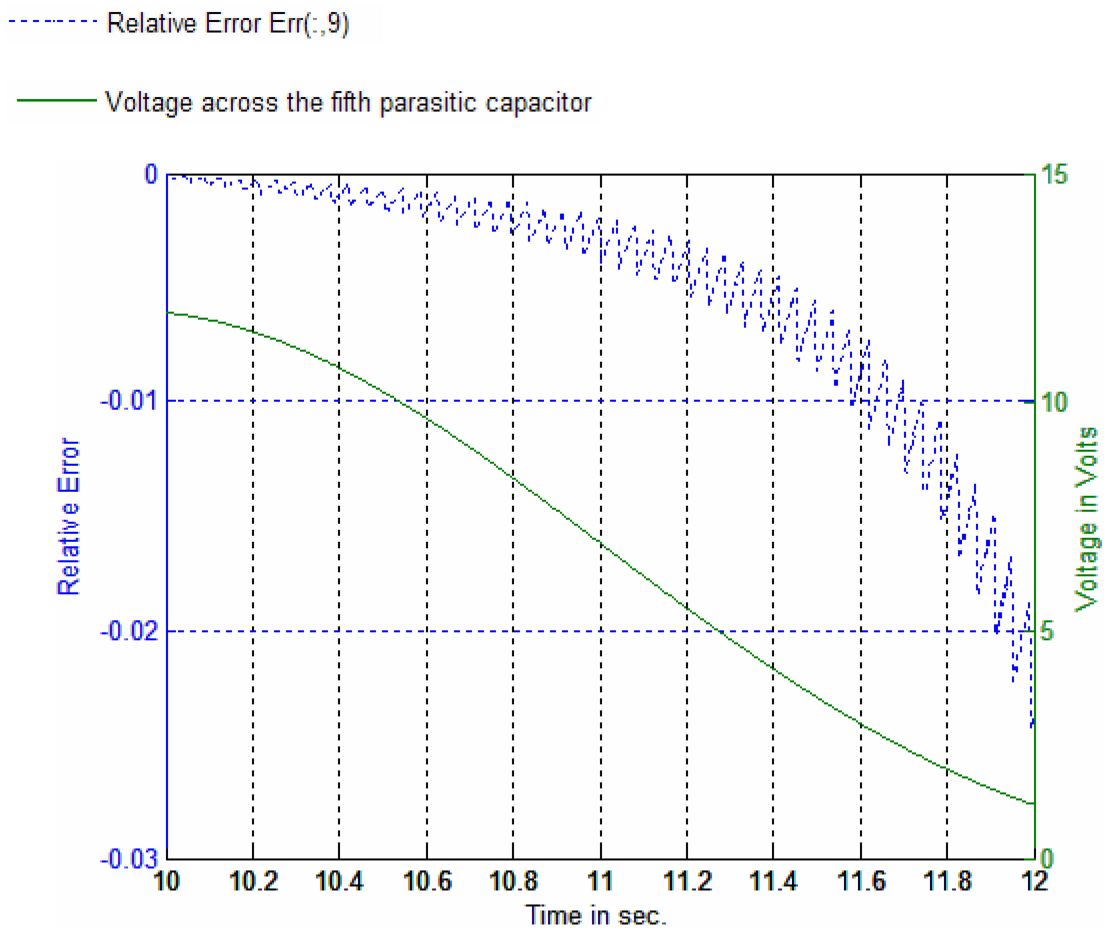

Figure 33- Voltage across the ninth capacitor and the corresponding relative error between for $10\text{sec.} \le T \le 12\text{sec.}$

As is reflected by the figure above, the relative error of the 9th state has a small magnitude in this interval:

$$-0.024329 \le \text{err}(5001:6001,9) \le -5.51697\text{e}-005,$$

and absolute mean percentage of:

$$|mean(err(5001:6001,9))| = 0.50413\%$$

Where $T(5001) = 10$ seconds and $T(6001) = 12$ seconds.

As stated for the previous study case, the spikes observed in Figure 32 are due to numerical discrepancy induced by the relative error calculation. For instance, if we consider the maximum of the relative error corresponding to the 5th state:

$$\max\left|(err(:,5))\right| = 258.8794$$

This occurs at index 6831 of the err matrix, i.e. $err(6831,5) = 258.8794$. Knowing that

$$err(i,j) = \frac{X_{n2}(i,j) - X_{opt}(i,j)}{X_{n2}(i,j)}, \text{ for } 1 \le i \le 10001 \text{ and } 1 \le i \le 14$$

Hence, $err(6831,5) = \dfrac{X_{n2}(6831,5) - X_{opt}(6831,5)}{X_{n2}(6831,5)}$, where

$X_{n2}(6831,5) = -3.27509698 \times 10^{-5}$ and $X_{opt}(6831,5) = 8.4458 \times 10^{-3}$. Both of these values have small magnitudes however one of them has order 10^{-5} and the other 10^{-3} which induced the large jump in the relative error magnitude. By setting a threshold value such that whenever the order of both X_{n2} and X_{opt} is less than or equal to 10^{-3} set the corresponding relative error value to zero. Hence, with the application of this constraint, the absolute maximums of the relative error become:

$$\max\left|(err(:,5))\right| = 6.0306,\ \max\left|(err(:,6))\right| = 0.4166,\ \max\left|(err(:,7))\right| = 0.0582,$$

$$\max\left|(err(:,8))\right| = 0.0239,\ \max\left|(err(:,9))\right| = 80.756.$$

with the following absolute mean percentages:

$$mean\left|(err(:,5))\right| = 0.0524\%,\ mean\left|(err(:,6))\right| = 0.0436\%,\ mean\left|(err(:,7))\right| = 0.1377\%,$$

$$mean\left|(err(:,8))\right| = 0.1121\%,\ mean\left|(err(:,9))\right| = 4.4930\%.$$

Again, the absolute maximum and percentage averages have been greatly decreased, which shows the effect of calculation discrepancies on the error especially in regions where the state variable values become minute when approaching zero.

5.3 Conclusions:

The main area of interest in the state trajectories is when the voltage across the $N+1$ capacitor reaches its maximum of $Vc'_{N+1} = N \cdot Vc'_i(0)$, where $i = \{1, 2, \cdots, N\}$ while the voltage across the remaining N parasitic capacitors is approaching zero. Using the state space generation algorithm we have explained in Chapter 2 to generate a new desired state trajectory X_n and the least-square nonlinear optimization algorithm to obtain the optimal set of parasitic capacitors to track the new X_n, we obtained X_{opt} whose state trajectories closely follow the desired X_n especially in the interval of interest. Adding a constraint to the relative error magnitude, such that whenever the value of the states go down below a certain threshold value set the corresponding relative error to zero, proved that X_{opt} closely follows X_n with negligible mean relative error value which undermines the effect of the sudden jumps in the relative error magnitude. It is worth noting here that the time shift in this section could have been reproduced by time scaling the state space model of the Marx generator

$$\dot{X}(t) = \frac{1}{fc_i} {}^{N}M \cdot X(t)$$

It is worth noting here that the time shift in this section could have been reproduced by time scaling the state space model of the Marx generator

$$\dot{X}(t) = \frac{1}{fc_i} {}^{N}M \cdot X(t)$$

However, even though this concept is applicable for the case of a Marx generator it is cannot be generalized for any state space mode. The goal here is to show that by shifting the state trajectories of a particular system it is still possible to determine the design parameters that will yield the desired output.

Chapter 6 Conclusion

We demonstrated in this thesis how to generate the appropriate state space model for any N stages Marx generator by simply following the algorithm developed in Chapter 2. This state space representation can be used to analyze Marx generators with arbitrary large number of stages and introduces the advantage of knowing the behavior of the voltages across the capacitors and the currents across the inductors at all instance of time and hence prevent any unpleasant behavioral surprise.

We developed in Chapter 4 an algorithm that exploits the state trajectory properties of periodicity and symmetry to construct a new set of states that comply with the user's desired specifications. Hence, a user can now specify the time at which he desires the voltage across the $N+1$ parasitic capacitor to peek, that is the time instant t_f at which $Vc'_{N+1}(t_f) = N \cdot Vc'_i(0)$, where $i = \{1, 2, \cdots, N\}$, and the algorithm generates the appropriate state trajectories that achieve that goal. This is where the application of the nonlinear least-squares algorithm comes into play. Up till now the new desired state trajectories are just theoretical and not the resulting output of the corresponding Marx generator. To be able to determine the parasitic capacitor values that produce the desired state behavior, we used a nonlinear least-squares optimization algorithm based on the quasi-Newton Levenberg-Marquardt algorithm with line search procedures.

This strategy was shown to be successful with small relative error between the state trajectories in X_{opt}, obtained from the new set of parasitic capacitors, and the new desired trajectories in X_n, generated using the algorithm of Chapter 2. At some particular instants of time, we encountered large jumps in the relative error magnitude. More precisely, the relative error peeks occur at instants of time when the voltages across the $N+1$ parasitic capacitors are approaching zero. These jumps can be undermined without any serious effect on the system's behavior; primarily because we are approximating the behavior of a Marx generator using a base reference model that is by no means ideal and because they do not occur at time intervals of interest that is the time intervals when the voltage across the $(N+1)^{st}$ parasitic capacitor is peeking while the remaining voltages and currents are approaching zero. In addition to the numerical discrepancies induced by the calculation of the relative error, these error magnitude jumps might be due to the optimization algorithm we are using that has proven to be less efficient when encountering line search inaccuracies. Hence, by setting a threshold value such that whenever the state value goes below that threshold the error is set to zero, we were able to eliminate all the spikes and hence obtain both relative error and mean error values that are negligible. It is worth noting that we generated the new sets of "desired" state trajectories based on a model that is an approximation of what the behavior of a Marx generator would be. Therefore, obtaining and basing our new state trajectories on a reference model that has proven to be more accurate can greatly improve our results as we will be attempting to track more realistic state trajectory models. Other issues worth looking at are using another variation of the optimization algorithm such as the Guass-Newton methods and predicting the state trajectories of an N stages generator using the trajectory models of $N-1, N-2, \cdots, 2$ stages Marx generators. Besides that, it would be interesting to apply the concepts in this thesis to an actual Marx generator and compare the simulation results to the practical results and, thus, obtain a better idea of the accuracy and effectiveness of the results obtained in this thesis.

Therefore by applying the concepts developed in this thesis to a Marx generator circuit, we have showed that by generating the state space model of any circuit in combination with the appropriate optimization algorithm, a user can specify a desired output trajectory of hi circuit at a particular instant of time and determine in return the design parameters that yield a trajectory that best tracks his output reference model.

As an interesting application of the concepts developed in this thesis, provided appropriate safety measures are taking into consideration, I propose providing the user with an interface through which he can specify the instant of time he desires the spark at the $N+1$ parasitic capacitor to occur and the number of stages he desires, this number would be dependant on the charging voltage sources available for the user and the maximum output voltage value he is seeking. The device’s memory will contain the state space trajectories for a wide range of stage numbers. The tool will then use the shifting algorithm of Chapter 4, the state space generation algorithm of Chapter 2 and the nonlinear least-squares optimization algorithm of Chapter 3 to generate the appropriate parasitic capacitor values that will simulate the state space trajectories that conform to the user’s specifications.

Appendix A:

A.1 N=4-Stage Marx Generator

Now computing the state space realization of an N=4 stages Marx generator:

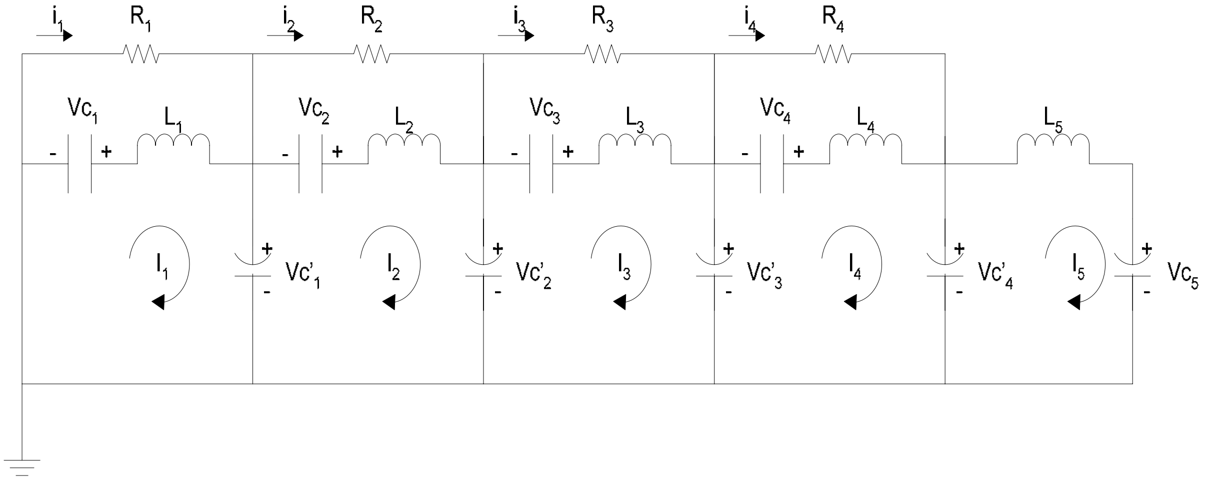

Figure 34- A N=4-Stage Marx Generator

Let

$$X_1 = Vc_1(t),\ X_2 = Vc_2(t),\ X_3 = Vc_3(t),\ X_4 = Vc_4(t),\ X_5 = Vc_1'(t),\ X_6 = Vc_2'(t),$$
$$X_7 = Vc_3'(t), X_8 = Vc_4'(t),\ X_9 = Vc_5'(t),\ X_{10} = I_1(t),\ X_{11} = I_2(t),\ X_{12} = I_3(t),$$
$$X_{13} = I_4(t),\ X_{14} = I_5(t).$$

Similarly to the 2-stage Marx generator, using graph theory we obtain the following graph

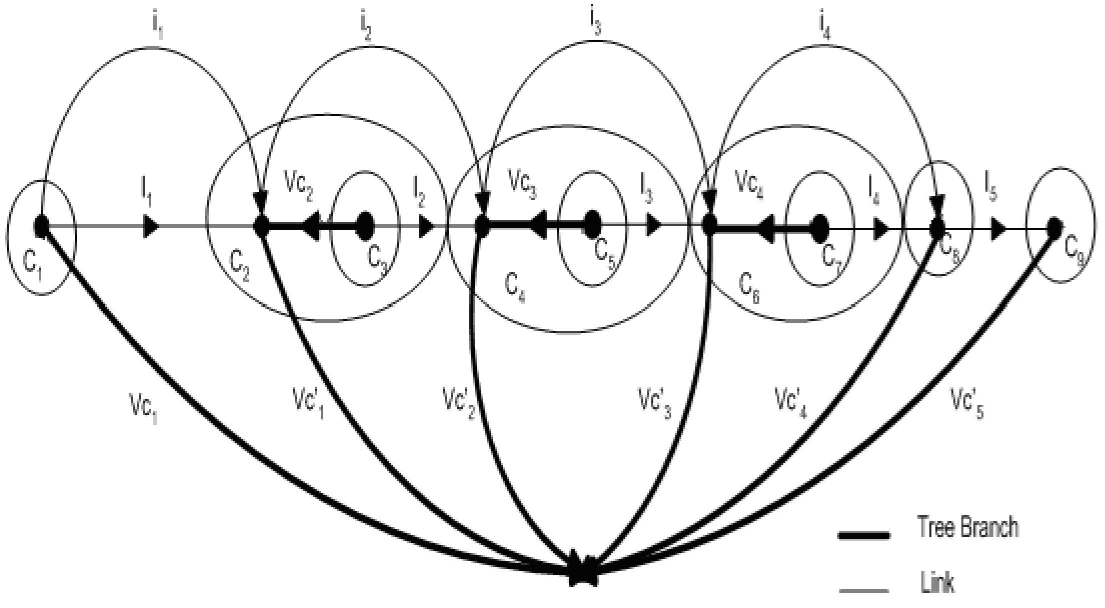

Figure 35- Graph representation of a 4-stage Marx generator

Having chosen the states to be voltages across capacitors and currents across inductors we follow these two simple rules stated in 93[4]:

3. Write KCL for every fundamental cut set (i.e. one tree branch and a number of links) in the network formed by each capacitor in the tree.
4. Write KVL for every fundamental loop (i.e. one link and a number of tree branches) in the network formed by each inductor in the co-tree (complement of a tree).

Cut set C₁: $C_1 \frac{dVc_1}{dt} + i_1 + I_1 = 0 \Rightarrow C_1\dot{X}_1 + i_1 + X_{10} = 0$ (A.1.1)

Cut set C₂: $-i_1 - I_1 + C_1^{'} \frac{dVc_1^{'}}{dt} + i_2 + I_2 = 0 \Rightarrow -i_1 - X_{10} + C_1^{'}\dot{X}_5 + i_2 + X_{11} = 0$ (A.1.5)

Cut set C₃: $C_2 \frac{dVc_2}{dt} + I_2 = 0 \Rightarrow C_2\dot{X}_2 + X_{11} = 0 \Rightarrow \dot{X}_2 = -\frac{1}{C_2}X_{11}$ (A.1.2)

Cut set C₄: $-i_2 - I_2 + C_2^{'} \frac{dVc_2^{'}}{dt} + i_3 + I_3 = 0 \Rightarrow -i_2 - X_{11} + C_2^{'}\dot{X}_6 + i_3 + X_{12} = 0$ (A.1.6)

Cut set C5: $C_3\frac{dVc_3}{dt}+I_3=0\Rightarrow C_3\dot{X}_3+X_{12}=0\Rightarrow \dot{X}_3=-\frac{1}{C_3}X_{12}$ (A.1.3)

Cut set C6: $-i_3-I_3+C_3^{'}\frac{dVc_3^{'}}{dt}+i_4+I_4=0\Rightarrow -i_3-X_{12}+C_3^{'}\dot{X}_7+i_4+X_{12}=0$ (A.1.7)

Cut set C7: $C_4\frac{dVc_4}{dt}+I_4=0\Rightarrow C_4\dot{X}_4+X_{13}=0\Rightarrow \dot{X}_4=-\frac{1}{C_4}X_{13}$ (A.1.4)

Cut set C7: $-i_4-I_4+C_4^{'}\frac{dVc_4^{'}}{dt}+I_5=0\Rightarrow -i_4-X_{13}+C_4^{'}\dot{X}_8+X_{14}=0$ (A.1.8)

Cut set C8: $-I_5+C_5^{'}\frac{dVc_5^{'}}{dt}=0\Rightarrow -X_{14}+C_5^{'}\dot{X}_9=0\Rightarrow \dot{X}_9=\frac{1}{C_5^{'}}X_{14}$ (A.1.9)

Loop 1 ($I_1\rightarrow Vc_1^{'}\rightarrow Vc_1$ **):**

$L_1\frac{dI_1}{dt}+Vc_1^{'}-Vc_1=0\Rightarrow L_1\dot{X}_{10}+X_5-X_1=0\Rightarrow \dot{X}_{10}=\frac{1}{L_1}X_1-\frac{1}{L_1}X_5$ (A.1.10)

Loop 2 ($I_1\rightarrow Vc_2^{'}\rightarrow Vc_1^{'}\rightarrow Vc_2$ **):**

$L_2\frac{dI_2}{dt}+Vc_2^{'}-Vc_1^{'}-Vc_2=0\Rightarrow L_2\dot{X}_{11}+X_6-X_5-X_2=0\Rightarrow$
$\dot{X}_{11}=\frac{1}{L_2}X_2+\frac{1}{L_2}X_5-\frac{1}{L_2}X_6$ (A.1.11)

Loop 3 ($I_3\rightarrow Vc_3^{'}\rightarrow Vc_2^{'}\rightarrow Vc_3$ **):**

$L_3\frac{dI_3}{dt}+Vc_3^{'}-Vc_2^{'}-Vc_3=0\Rightarrow L_3\dot{X}_{12}+X_7-X_6-X_3=0\Rightarrow$
$\dot{X}_{12}=\frac{1}{L_3}X_3+\frac{1}{L_3}X_6-\frac{1}{L_3}X_7$ (A.1.12)

Loop 4 ($I_4\rightarrow Vc_4^{'}\rightarrow Vc_3^{'}\rightarrow Vc_4$ **):**

$L_4\frac{dI_4}{dt}+Vc_4^{'}-Vc_3^{'}-Vc_4=0\Rightarrow L_4\dot{X}_{13}+X_8-X_7-X_4=0\Rightarrow$
$\dot{X}_{13}=\frac{1}{L_4}X_4+\frac{1}{L_4}X_7-\frac{1}{L_4}X_8$ (A.1.13)

Loop 5 ($I_5 \rightarrow Vc_5^{'} \rightarrow Vc_4^{'}$ **):**

$$L_5 \frac{dI_5}{dt} + Vc_5^{'} - Vc_4^{'} = 0 \Rightarrow L_5 \dot{X}_{14} + X_9 - X_8 = 0 \Rightarrow \dot{X}_{14} = \frac{1}{L_5} X_8 - \frac{1}{L_5} X_9 \text{ (A.1.14)}$$

Eliminating i_1, i_2, i_3, i_4 :

Loop 6 ($i_1 \rightarrow Vc_1^{'} \rightarrow Vc_1$ **):**

$$R_1 i_1 + Vc_1^{'} - Vc_1 = 0 \Rightarrow R_1 i_1 + X_5 - X_1 = 0 \Rightarrow i_1 = \frac{1}{R_1}(X_1 - X_5) \text{ (A.1.15)}$$

Loop 7 ($i_2 \rightarrow Vc_2^{'} \rightarrow Vc_1^{'}$ **):**

$$R_2 i_2 + Vc_2^{'} - Vc_1^{'} = 0 \Rightarrow R_2 i_2 + X_6 - X_5 = 0 \Rightarrow i_2 = \frac{1}{R_2}(X_5 - X_6) \text{ (A.1.16)}$$

Loop 8 ($i_3 \rightarrow Vc_3^{'} \rightarrow Vc_2^{'}$ **):**

$$R_3 i_3 + Vc_3^{'} - Vc_2^{'} = 0 \Rightarrow R_3 i_3 + X_7 - X_6 = 0 \Rightarrow i_3 = \frac{1}{R_3}(X_6 - X_7) \text{ (A.1.17)}$$

Loop 9 ($i_4 \rightarrow Vc_4^{'} \rightarrow Vc_3^{'}$ **):**

$$R_4 i_4 + Vc_4^{'} - Vc_3^{'} = 0 \Rightarrow R_4 i_4 + X_8 - X_7 = 0 \Rightarrow i_4 = \frac{1}{R_4}(X_7 - X_8) \text{ (A.1.18)}$$

Replacing (A.1.15) in (A.1.1) we obtain:

$$C_1 \dot{X}_1 + \frac{1}{R_1} X_1 - \frac{1}{R_1} X_5 + X_{10} = 0 \Rightarrow \dot{X}_1 = -\frac{1}{R_1 C_1} X_1 + \frac{1}{R_1 C_1} X_5 - \frac{1}{C_1} X_{10} \text{ (A.1.1)}$$

Replacing (A.1.15) and (A.1.16) in (A.1.5) we obtain:

$$-\frac{1}{R_1} X_1 + \frac{1}{R_1} X_5 - X_{10} + C_1^{'} \dot{X}_5 + \frac{1}{R_2} X_5 - \frac{1}{R_2} X_6 + X_{11} = 0$$

$$\Rightarrow -\frac{1}{R_1} X_1 + \frac{R_1 + R_2}{R_1 R_2} X_5 - \frac{1}{R_2} X_6 - X_{10} + X_{11} + C_1^{'} \dot{X}_5 = 0$$

$$\dot{X}_5 = \frac{1}{R_1 C_1^{'}} X_1 - \frac{R_1 + R_2}{R_1 R_2 C_1^{'}} X_5 + \frac{1}{R_2 C_1^{'}} X_6 + \frac{1}{C_1^{'}} X_{10} - \frac{1}{C_1^{'}} X_{11} \text{ (A.1.5)}$$

Replacing (A.1.16) and (A.1.17) in (A.1.6) we obtain:

$$-\frac{1}{R_2}X_5+\frac{1}{R_2}X_6-X_{11}+C_2^{'}\dot{X}_6+\frac{1}{R_3}X_6-\frac{1}{R_3}X_7+X_{12}=0$$

$$\Rightarrow -\frac{1}{R_2}X_5+\frac{R_2+R_3}{R_2R_3}X_6-\frac{1}{R_3}X_7-X_{11}+X_{12}+C_2^{'}\dot{X}_6=0$$

$$\dot{X}_6=\frac{1}{R_2C_2^{'}}X_5-\frac{R_2+R_3}{R_2R_3C_2^{'}}X_6+\frac{1}{R_3C_2^{'}}X_7+\frac{1}{C_2^{'}}X_{11}-\frac{1}{C_2^{'}}X_{12}\quad\text{(A.1.6)}$$

Replacing (A.1.17) and (A.1.18) in (A.1.7) we obtain:

$$-\frac{1}{R_3}X_6+\frac{1}{R_3}X_7-X_{12}+C_3^{'}\dot{X}_7+\frac{1}{R_4}X_7-\frac{1}{R_4}X_8+X_{13}=0$$

$$\Rightarrow -\frac{1}{R_3}X_6+\frac{R_3+R_4}{R_3R_4}X_7-\frac{1}{R_4}X_8-X_{12}+X_{13}+C_3^{'}\dot{X}_7=0$$

$$\dot{X}_7=\frac{1}{R_3C_3^{'}}X_6-\frac{R_3+R_4}{R_3R_4C_3^{'}}X_7+\frac{1}{R_4C_3^{'}}X_8+\frac{1}{C_3^{'}}X_{12}-\frac{1}{C_3^{'}}X_{13}\quad\text{(A.1.7)}$$

Replacing (A.1.18) in (A.1.8) we obtain:

$$-\frac{1}{R_4}X_7+\frac{1}{R_4}X_8-X_{13}+X_{14}+C_4^{'}\dot{X}_8=0$$

$$\dot{X}_8=\frac{1}{R_4C_4^{'}}X_7-\frac{1}{R_4C_4^{'}}X_8+\frac{1}{C_4^{'}}X_{13}-\frac{1}{C_4^{'}}X_{14}\quad\text{(A.1.8)}$$

Now we have the following set of equations:

$$\dot{X}_1=-\frac{1}{R_1C_1}X_1+\frac{1}{R_1C_1}X_5-\frac{1}{C_1}X_{10}$$

$$\dot{X}_2=-\frac{1}{C_2}X_{11}$$

$$\dot{X}_3=-\frac{1}{C_3}X_{12}$$

$$\dot{X}_4=-\frac{1}{C_4}X_{13}$$

$$\dot{X}_5 = \frac{1}{R_1C_1^{'}}X_1 - \frac{R_1+R_2}{R_1R_2C_1^{'}}X_5 + \frac{1}{R_2C_1^{'}}X_6 + \frac{1}{C_1^{'}}X_{10} - \frac{1}{C_1^{'}}X_{11}$$

$$\dot{X}_6 = \frac{1}{R_2C_2^{'}}X_5 - \frac{R_2+R_3}{R_2R_3C_2^{'}}X_6 + \frac{1}{R_3C_2^{'}}X_7 + \frac{1}{C_2^{'}}X_{11} - \frac{1}{C_2^{'}}X_{12}$$

$$\dot{X}_7 = \frac{1}{R_3C_3^{'}}X_6 - \frac{R_3+R_4}{R_3R_4C_3^{'}}X_7 + \frac{1}{R_4C_3^{'}}X_8 + \frac{1}{C_3^{'}}X_{12} - \frac{1}{C_3^{'}}X_{13}$$

$$\dot{X}_8 = \frac{1}{R_4C_4^{'}}X_7 - \frac{1}{R_4C_4^{'}}X_8 + \frac{1}{C_4^{'}}X_{13} - \frac{1}{C_4^{'}}X_{14}$$

$$\dot{X}_9 = \frac{1}{C_5^{'}}X_{14}$$

$$\dot{X}_{10} = \frac{1}{L_1}X_1 - \frac{1}{L_1}X_5$$

$$\dot{X}_{11} = \frac{1}{L_2}X_2 + \frac{1}{L_2}X_5 - \frac{1}{L_2}X_6$$

$$\dot{X}_{12} = \frac{1}{L_3}X_3 + \frac{1}{L_3}X_6 - \frac{1}{L_3}X_7$$

$$\dot{X}_{13} = \frac{1}{L_4}X_4 + \frac{1}{L_4}X_7 - \frac{1}{L_4}X_8$$

$$\dot{X}_{14} = \frac{1}{L_5}X_8 - \frac{1}{L_5}X_9$$

Hence, we can now write our state space representation of the form:

$$\dot{X} = {}^4M \cdot X \ ,$$

Where ${}^4M = \begin{bmatrix} {}^4M_{11} & {}^4M_{12} \\ {}^4M_{21} & {}^4M_{22} \end{bmatrix}$ is a 14×14 matrix and X is a 1x14 column vector.

${}^{4}M_{11}$ is a 9x9 matrix and has the following structure

$-\frac{1}{RC}$	0	0	0	$\frac{1}{RC}$	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
$\frac{1}{RC_1^{'}}$	0	0	0	$-\frac{(R+R)}{RRC_1^{'}}$	$\frac{1}{RC_1^{'}}$	0	0	0
0	0	0	0	$\frac{1}{RC_2^{'}}$	$-\frac{(R+R)}{RRC_2^{'}}$	$\frac{1}{RC_2^{'}}$	0	0
0	0	0	0	0	$\frac{1}{RC_3^{'}}$	$-\frac{(R+R)}{RRC_3^{'}}$	$\frac{1}{RC_3^{'}}$	0
0	0	0	0	0	0	$\frac{1}{RC_4^{'}}$	$-\frac{1}{RC_4^{'}}$	0
0	0	0	0	0	0	0	0	0

Table 5- ${}^{4}M_{11}$ matrix

$^4M_{21}$ is a 6x9 matrix and has the following structure:

$\frac{1}{L}$	0	0	0	$-\frac{1}{L}$	0	0	0	0
0	$\frac{1}{L}$	0	0	$\frac{1}{L}$	$-\frac{1}{L}$	0	0	0
0	0	$\frac{1}{L}$	0	0	$\frac{1}{L}$	$-\frac{1}{L}$	0	0
0	0	0	$\frac{1}{L}$	0	0	$\frac{1}{L}$	$-\frac{1}{L}$	0
0	0	0	0	0	0	0	$\frac{1}{L_5}$	$-\frac{1}{L_5}$

Table 6- $^4M_{21}$ matrix

$^4M_{12}$ is a 9x5 matrix and has the following structure:

$-\frac{1}{C}$	0	0	0	0
0	$-\frac{1}{C}$	0	0	0
0	0	$-\frac{1}{C}$	0	0
0	0	0	$-\frac{1}{C}$	0
$\frac{1}{C_1'}$	$-\frac{1}{C_1'}$	0	0	0
0	$\frac{1}{C_2'}$	$-\frac{1}{C_2'}$	0	0

0	0	$\frac{1}{C_3^{'}}$	$-\frac{1}{C_3^{'}}$	0
0	0	0	$\frac{1}{C_4^{'}}$	$-\frac{1}{C_4^{'}}$
0	0	0	0	$\frac{1}{C_5^{'}}$

Table 7- ${}^{4}M_{12}$ matrix

${}^{4}M_{22}$ is a 5x5 matrix and has the following structure:

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Table 8- ${}^{4}M_{22}$ matrix

Please note that the $N = 4$ stages Marx generator structure that we are presenting conforms to all the constraints set in Chapter 2 for a general N stages Marx generator and has, similarly to the $N = 2$ stages Marx generator, the following parameter values:

$C = C_1 = C_2 = C_3 = C_4 = 1F$,

$L = L_1 = L_2 = L_3 = L_4 = 1H$,

$R = R_1 = R_2 = R_3 = R_4 = 100,000\Omega$,

$L_5 = N \times L = 4 \times 1 = 4H$,

$C_1^{'} = 0.03599F, C_2^{'} = 0.067215F, C_3^{'} = 0.02363F, C_4^{'} = 0.12468F$,

$$C_5^{'} = \frac{C}{N} = \frac{1}{4} F$$

The simulation results for this system were obtained by initially setting $I_{1,2,3,4,5}(0) = 0A$, $Vc'_{1,2,3,4,5}(0) = 0V$ and $Vc_{1,2,3,4}(0) = 3V$, which represents the voltage value to which the C_1, C_2, C_3, C_4 capacitors were initially charged:

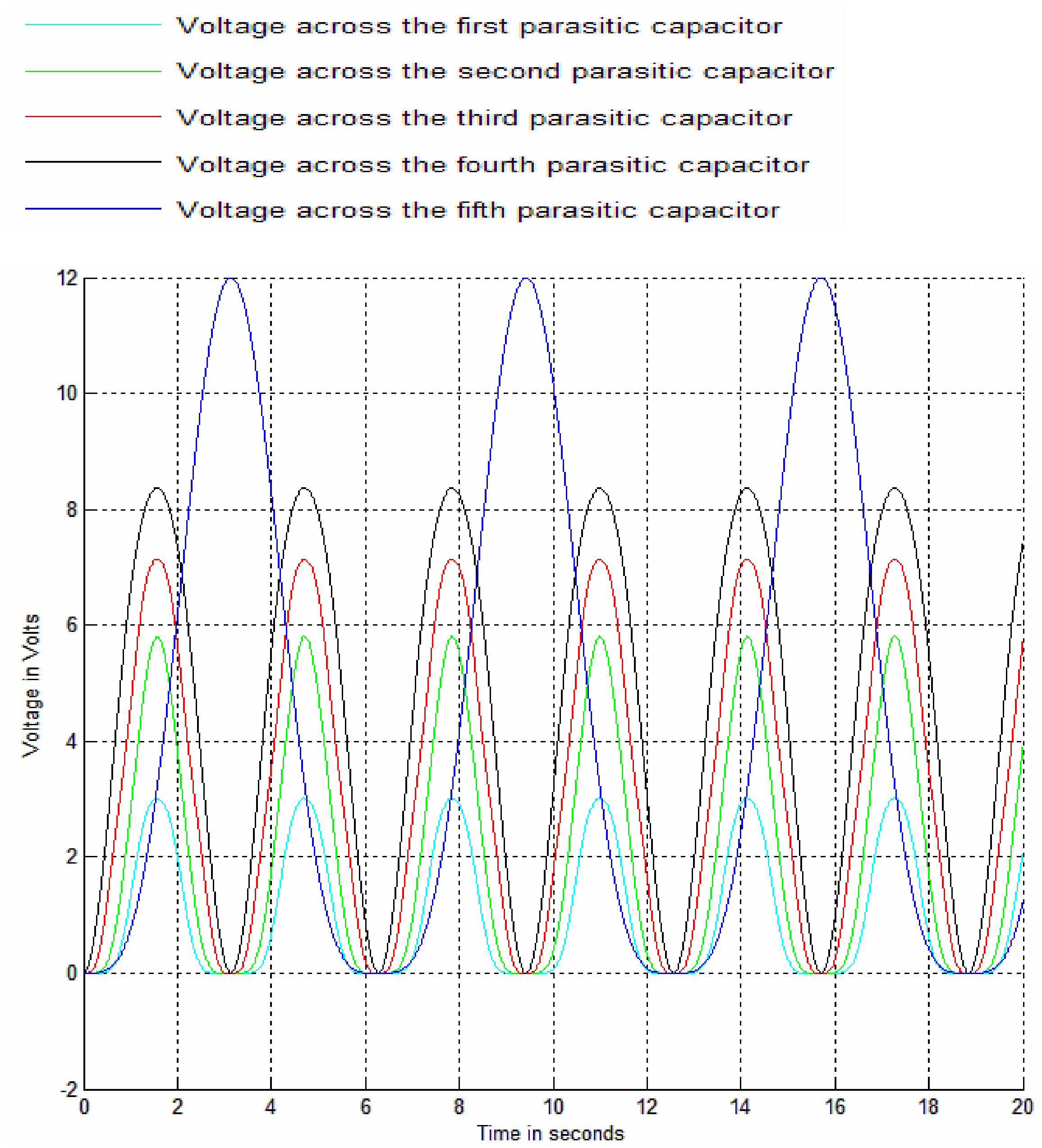

Figure 36-State trajectory representing the voltage across the parasitic capacitors of a 4-stage Marx generator

Similarly to the N=2-stage Marx generator, at the instant of time $t_f = 3.142$ seconds, the voltage across the first four parasitic capacitors, $Vc_1^{'}, Vc_2^{'}, Vc_3^{'}, Vc_4^{'}$, is approximately zero, and the voltage across the fifth parasitic capacitor, $Vc_5^{'}(t_f) = N \cdot Vc_{1,2,3,4}(0) = 4 \cdot 3 = 12V$.

A.2 N=8-Stage Marx Generator

Using the algorithm developed in Chapter 2 we can generate the corresponding

${}^{8}M$ matrix for an N=8-stages Marx generator.

First we know that the number of states expected from an N=8 stages Max generator would be $S = 3N + 2 = 3 \cdot 8 + 2 = 26$, therefore our ${}^{8}M = \begin{bmatrix} {}^{8}M_{11} & {}^{8}M_{12} \\ {}^{8}M_{21} & {}^{8}M_{22} \end{bmatrix}$ matrix would have to be a 26×26 matrix.

If we start by considering ${}^{8}M_{11}$, we know that it will have a size of $(2N+1)\times(2N+1)$, that is 17×17 with the following structure:

${}^{8}M_{11} = \begin{bmatrix} {}^{11}m_{11} & {}^{11}m_{12} \end{bmatrix}$, where ${}^{11}m_{11}$ and ${}^{11}m_{12}$ are respectively 17×8 and 17×9 matrices:

$^{11}m_{11}$ looks as follows:

$-\frac{1}{R_1C_1}$	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
$\frac{1}{R_1C_1}$	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Table 9- $^{11}m_{11}$ matrix

Due to the large number of entries in ${}^{11}m_{12}$, it will be expressed as ${}^{11}m_{12} = [a \quad b]$, where a and b are respectively 17×4 and 17×5 matrices.

Hence, a has the following structure:

$\frac{1}{R_1C_1}$	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
$-\frac{R_1+R_2}{R_1R_2C_1^{'}}$	$\frac{1}{R_2C_1^{'}}$	0	0
$\frac{1}{R_2C_2^{'}}$	$-\frac{R_2+R_3}{R_2R_3C_2^{'}}$	$\frac{1}{R_3C_2^{'}}$	0
0	$\frac{1}{R_3C_3^{'}}$	$-\frac{R_3+R_4}{R_3R_4C_3^{'}}$	$\frac{1}{R_4C_3^{'}}$
0	0	$\frac{1}{R_4C_4^{'}}$	$-\frac{R_4+R_5}{R_4R_5C_4^{'}}$
0	0	0	$\frac{1}{R_5C_5^{'}}$
0	0	0	0
0	0	0	0

0	0	0	0
0	0	0	0

Table 10- a matrix such that ${}^{11}m_{12} = \left[a \quad b\right]$

And b the following structure:

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
$\frac{1}{R_5C_4^{'}}$	0	0	0	0
$-\frac{R_5+R_6}{R_5R_6C_5^{'}}$	$\frac{1}{R_6C_5^{'}}$	0	0	0
$\frac{1}{R_6C_6^{'}}$	$-\frac{R_6+R_7}{R_6R_7C_6^{'}}$	$\frac{1}{R_7C_6^{'}}$	0	0
0	$\frac{1}{R_7C_7^{'}}$	$-\frac{R_7+R_8}{R_7R_8C_7^{'}}$	$\frac{1}{R_8C_7^{'}}$	0

0	0	$\frac{1}{R_8 C_8^{'}}$	$-\frac{1}{R_8 C_8^{'}}$	0
0	0	0	0	0

Table 11- b matrix such that ${}^{11}m_{12} = \left[a \quad b\right]$

Similarly if we now consider ${}^{8}M_{12}$, it will have be a $\left(2N+1\right)\times\left(N+1\right)$ matrix, that is 17×9 with the following structure:

$-\frac{1}{C_1}$	0	0	0	0	0	0	0	0
0	$-\frac{1}{C_2}$	0	0	0	0	0	0	0
0	0	$-\frac{1}{C_3}$	0	0	0	0	0	0
0	0	0	$-\frac{1}{C_4}$	0	0	0	0	0
0	0	0	0	$-\frac{1}{C_5}$	0	0	0	0
0	0	0	0	0	$-\frac{1}{C_6}$	0	0	0
0	0	0	0	0	0	$-\frac{1}{C_7}$	0	0
0	0	0	0	0	0	0	$-\frac{1}{C_8}$	0
$\frac{1}{C_1^{'}}$	$-\frac{1}{C_1^{'}}$	0	0	0	0	0	0	0

0	$\frac{1}{C_2^{'}}$	$-\frac{1}{C_2^{'}}$	0	0	0	0	0	0
0	0	$\frac{1}{C_3^{'}}$	$-\frac{1}{C_3^{'}}$	0	0	0	0	0
0	0	0	$\frac{1}{C_4^{'}}$	$-\frac{1}{C_4^{'}}$	0	0	0	0
0	0	0	0	$\frac{1}{C_5^{'}}$	$-\frac{1}{C_5^{'}}$	0	0	0
0	0	0	0	0	$\frac{1}{C_6^{'}}$	$-\frac{1}{C_6^{'}}$	0	0
0	0	0	0	0	0	$\frac{1}{C_7^{'}}$	$-\frac{1}{C_7^{'}}$	0
0	0	0	0	0	0	0	$\frac{1}{C_8^{'}}$	$-\frac{1}{C_8^{'}}$
0	0	0	0	0	0	0	0	$\frac{1}{C_9^{'}}$

Table 12- $^{8}M_{12}$ matrix

$^{8}M_{21}$’s size is $(N+1)\times(2N+1)$, i.e. it is a 9×17 matrix with the following structure:

$^{8}M_{21}=\left[{}^{21}m_{11} \quad {}^{21}m_{12}\right]$, where $^{21}m_{11}$ and $^{21}m_{12}$ are respectively 9×8 and 9×9 matrices.

$^{21}m_{11}$ has the following structure:

$\frac{1}{L_1}$	0	0	0	0	0	0	0
0	$\frac{1}{L_2}$	0	0	0	0	0	0

0	0	$\frac{1}{L_3}$	0	0	0	0	0
0	0	0	$\frac{1}{L_4}$	0	0	0	0
0	0	0	0	$\frac{1}{L_5}$	0	0	0
0	0	0	0	0	$\frac{1}{L_6}$	0	0
0	0	0	0	0	0	$\frac{1}{L_7}$	0
0	0	0	0	0	0	0	$\frac{1}{L_8}$
0	0	0	0	0	0	0	0

Table 13- ${}^{21}m_{11}$ matrix

${}^{21}m_{12}$ has the following structure:

$-\frac{1}{L_1}$	0	0	0	0	0	0	0	0
$\frac{1}{L_2}$	$-\frac{1}{L_2}$	0	0	0	0	0	0	0
0	$\frac{1}{L_3}$	$-\frac{1}{L_3}$	0	0	0	0	0	0
0	0	$\frac{1}{L_4}$	$-\frac{1}{L_4}$	0	0	0	0	0
0	0	0	$\frac{1}{L_5}$	$-\frac{1}{L_5}$	0	0	0	0

0	0	0	0	$\frac{1}{L_6}$	$-\frac{1}{L_6}$	0	0	0
0	0	0	0	0	$\frac{1}{L_7}$	$-\frac{1}{L_7}$	0	0
0	0	0	0	0	0	$\frac{1}{L_8}$	$-\frac{1}{L_8}$	0
0	0	0	0	0	0	0	$\frac{1}{L_9}$	$-\frac{1}{L_9}$

Table 14- $^{21}m_{12}$ matrix

The last part of the ^{8}M matrix is the $^{8}M_{22}$ block which has a size of $(N+1)\times(N+1)$, therefore it is a 9×9 matrix full of zeros:

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Table 15- $^{8}M_{22}$ matrix

References:

[1] EarthLink, “Marx Generators”, http://home.earthlink.net/~jimlux/hv/marx.htm

[2] Information Unlimited, “Marx Impulse Generators”, Accessed on 04/26/2006, http://www.amazing1.com/marx_generators.htm

[3] Andreas Kuthi, Ray Alde, Martin Gundersen and Andreas Neuber, “Marx Generator Using PseudoSpark Switches” .

[4] S. Karni, “Intermediate Network Analysis”, Allyn and Bacon Inc. Boston, 1971.

[5] The MathWorks, “Optimization Toolbox for Use with MatLab”, version 3.

[6] P. Dorato, C. Abdallah, V. Cerone, “Linear Quadratic Control an Introduction”.

[7] Broyden, C.G. “The Convergence of a Class of Double-Rank Minimzation Algorithms”, *J. Inst. Maths. Applics.*, Vol.6, pp 76-90, 1970.

[8] Fletcher, R. “A New Approach to Variable Metric Algorithms”, *Computer Journal*, Vol. 13, pp 317-322, 1970.

[9] Goldfarb, D. “A Family of Variable Metric Updates Derived by Variational Means”, *Mathematics of Computing*, Vol. 24, pp 23-26, 1970.

[10] Shanno, D.F., “Conditioning of Quasi-Newton Methods for Function Minimization”, *Mathematics of Computing*, Vol. 24, pp 647-656, 1970.

[11] Davidon, W.C. “Variable Metric Method for Minimization”, *A.E.C. Research and Development Report*, ANL-5990, 1959.

[12] Fletcher, R. and M.J.D. Powell, “A Rapidly Convergent Descent Method for Minimization”, *Computer Journal*, Vol. 6, pp 163-168, 1963.

[13] Georgia Institute of Technology System Realization Laboratory, “Quazi-Newton Methods”, http://www.srl.gatech.edu/education/ME6103/Quasi-Newton.ppt.