2010

Steady state analysis of oscillators

Elhassan, Ibrahim

Library and Archives
Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-71758-5
Our file Notre référence
ISBN: 978-0-494-71758-5

STEADY STATE ANALYSIS OF OSCILLATORS

By

Ibrahim Elhassan

A Thesis

Presented to Lakehead University

in Partial Fulfillment of the Requirements for the Degree of Master of Science

in

Electrical Engineering

Thunder Bay, Ontario

April 2010

Abstract

STEADY STATE ANALYSIS OF OSCILLATORS (under the supervision of Dr. Carlos Christoffersen)

A common method used for steady-state analysis of oscillators is called Harmonic Balance. Harmonic balance finds the steady-state solution directly in frequency-domain. However, Harmonic Balance is very sensitive to the initial guess and may not converge if the oscillation frequency is not known *a priori*. Sometimes it may converge to the unstable DC operating point of the oscillator. Moreover, it is usually difficult to have such good initial guess.

In this thesis, a fast approach is developed to improve the initial guess for Harmonic Balance (HB). This approach is derived from Minimal Polynomial Extrapolation (MPE) and Warped Multi- time Partial Differential Equation (WaMPDE). The WaMPDE works by separating the fast and slow variations in the response of oscillators, thus minimizing time and CPU consumption. The role of MPE is to accelerate the work of WaMPDE. The advantage of the MPE method is that it saves Jacobian matrix decomposition and it is easy to implement. Simulation results of different oscillators (Colpitts and LC-tuned bipolar) are presented to evaluate the performance of the proposed method.

Acknowledgment

I have had the privilege to work under Dr. Carlos Christoffersen at Lakehead University, his encouragement, perpetual energy, guidance and support from the initial to the final level helped me greatly during my the research. In addition, he was always accessible and willing to help. I am heartily thankful to him.

I also want to thank Dr. Xiaoping Liu, Dr. Abdelhamid Tayebi , N. Yu and Dr. A. Manzak for their help during my study. I would like to thank my brother Dr. E. Mohamedelhassan for his guidance and continuous inspiration. I dedicate this work to my late father Mr. Eltahir Mohamedelhassan.

TABLE OF CONTENTS

List of Figures

List of Tables

List of Symbols

t_1	Fast time
t_2	Slow time
τ_1	Warped time
τ_2	Real time
$\omega(\tau_2)$	Local angular Frequency
$x(t)$	State variable vector in time domain
$\boldsymbol{X}$	State variable vector in frequency domain
X_j^i	Fourier coefficient of the i^{th} harmonic of the j^{th} state variable
$x(\tau_1, \tau_2)$	State variable vector in warped multi-time domain
G	Conductance matrix
C	Linear charge matrix
$\in$	Tolerance
D	Tolerance dividing factor
$\boldsymbol{q}$	Vector of nonlinear charge in time-domain for all state variable
$\boldsymbol{Q}$	Vector of nonlinear charge in frequency-domain for all state variables
$\boldsymbol{f}$	Vector of nonlinear function in time domain for all state variables
$\boldsymbol{F}$	Vector of nonlinear function in frequency domain for all state variables
S(t)	Source vector
I_S	Saturation current of a transistor or a diode
V_T	Thermal voltage
i_s	Excitation current
C_j	Diode conjunction capacitance
v	Number of nodal voltages
a	Number of harmonics
p	Time step increase factor for each run.
n_{max}, n_{min}	Newton iterations constraints for adaptive time step algorithm
F	Time step factor for adaptive time step algorithm

Chapter 1

Introduction

1.1 Motivations of This Study

Oscillators are nonlinear circuits which generate periodic, time-varying waveforms from DC inputs. They are essential components in many electrical and mechanical systems. However, it is very challenging to analyze oscillators in a satisfactory and reliable manner. Previous analysis of oscillators depended on a practical design perspective that relied on linear models to obtain simple design formulas. However, linear models aren't really adequate for practical oscillators, since nonlinearity is essential for their orbital stability. Today numerical simulation represents the main tool in oscillator's analysis.

One of the most important aspects of oscillators is their steady-state. The steady-state analysis can be accomplished by methods in time-domain, frequency-domain or in a combination of both. The traditional time-domain method calculates the steady-state response by integrating the differential equations that describe the circuit from some chosen initial state until any transient behaviour dies out. This approach suffers from several drawbacks. One of these drawbacks is that the transient might take a long time to decay. Thus, this approach would involve expensive calculations. Several time-domain methods like the Shooting method [11, 28], the time-domain envelope-following method [14, 17, 18] and Extrapolation method [19,29] have been developed to address these drawbacks.

Harmonic Balance (HB) [30] method bypasses the transient and computes the steady-state directly. Each state variable is represented by a truncated Fourier series that satisfies the requirements of periodicity. The Fourier coefficients are calculated to satisfy the nonlinear equations. Despite its dominance, HB experiences some limitations. For example, HB needs a good initial guess to converge to the desired solution. If such good initial guess is not provided, HB may converge to the unstable DC operating point of the oscillator or may not converge at all. It is usually difficult to have such good initial guess. Another problem with the regular HB is that it becomes computationally expensive if the number of state variables that must be optimized is large.

Several methods are developed to deal with the limitations of HB. In particular, the method in Ref. [6] has improved the initial guess for the HB. The method is derived from Warped Multi-time Partial Differential Equation (WaMPDE) [1, 5] formulation. The WaMPDE works by separating the fast (oscillations) and slow variations (envelope) in the response of oscillators. With this separation, while the time constant of the slow variation determines the interval of computations, the fast variation rate is no longer restricting the integration step size. Thus the computational cost is minimized. The methods in Refs. [5,6] adaptively control the number of harmonics for each state variable, this allows reducing the number of harmonics and thus reducing the computational effort.

Computing the steady state (if exist) is finding the limit of the solution sequences of the transients. One way to accelerate their limit (the decay of the transient evolution) is to apply vector extrapolation methods. Extrapolation methods were proved to be efficient and they are very easy to implement [19]. The methods in Refs. [19,29] improved calculating the steady-state of oscillators in time-domain analysis.

In this work we made some improvements to capture the steady-state response of oscillators in frequency-domain. The attractiveness of the methods in Refs. [6,19] made them to be the base of our sought improvements. WaMPDE and adaptive HB are used to improve the initial guess and reduce computational effort respectively. Our improvement shall accelerate the decay of the transient calculated by WaMPDE in Ref. [6]. A vector extrapolation method known as Minimal Extrapolation method (MPE) [8,9,19,29] is used to accomplish our objective. Our approach accelerates the work of WaMPDE, therefore, reducing the computational effort.

1.2 Thesis Outline

Chapter 2 provides the literature review of the topics relevant to this thesis; oscillator theory, HB, multidimensional methods, the method in Ref. [6] and the MPE method in Refs. [8,9,19,29]. Chapter 3 presents the proposed approach to capture the steady- state of oscillators. Chapter 4 provides simulations of actual oscillators to test the proposed method and compares them to the simulations in Ref. [6]. Chapter 5 provides a summary of the simulations results and suggest some directions for future research.

Chapter 2

Literature Review

2.1 Introduction

Capturing the steady-state of oscillators is achieved by methods in time–domain, frequency-domain or in a combination of both. Traditional time-domain methods obtain the steady-state by integrating the nonlinear differential equations from an appropriate initial value until the transients have died out. If the transient has disappeared after a few periods, this is the simplest and most efficient approach for the computation of the steady-state solution. However, in many cases, the transients change very slowly, which often imply excessive integration before the steady-state solution is reached. Many methods have been developed to accelerate the decay of the transients. We will mention a few: the Shooting method [11,28] which finds the steady-state by iteratively integrating the ODEs over one period. The Time-domain Envelope-Following method [14,17,18] which computes a few transient cycles and then integrates the envelope to accelerate capturing the steady-state. One final example which is relevant to the work in this thesis is the Extrapolation methods [19,29] which calculate the steady-state by accelerating the decay of the transients using vector extrapolations.

The frequency-domain Harmonic Balance (HB) method calculates the steady-state of oscillators directly by avoiding the transient. Avoiding the transient usually results in reducing the simulation time in comparison with time-domain methods. Many techniques have been developed to address the limitations of HB that mentioned in Chapter 1. For example the techniques in Refs. [2,3] use only the necessary number of harmonics for each state variable which result in reducing the computational effort. One last example which is relevant to the work in this thesis is the method in Ref. [6], this method uses WaMPDE [1,5] and adaptive HB [5]. The WaMPDE is a multi-time scales method; it works by separating the fast and slow variations in the response of oscillators. The method in Ref. [6] uses adaptive HB, unlike regular HB; starting with small number of harmonics for all state variables and then each state variable adds more harmonics as needed without directly increasing the number of harmonics in the other state variables. The method in Ref. [6] managed to improve the speed and CPU computations.

The main focus of this chapter is to review the concepts and techniques that are relevant to the proposed method. The chapter is organized as follows: Section 2.2 gives an introduction to oscillator theory. Section 2.3 explains the concept of HB. Section 2.4 describes the concept of multidimensional methods. Section 2.5 illustrates how to use WaMPDE to model a simple oscillator. Section 2.6 describes the algorithm of the method in Ref. [6]. Section 2.7 explains MPE, which is applied to the method in Ref. [6] in order to improve it.

2.2 Oscillator Theory

An electronic oscillator is a circuit with periodic output signal but with no periodic input signal. It operates by converting dc power to periodic output signal (ac). Oscillators can be seen in many communication and electrical systems. Oscillators are inherently nonlinear circuits, but linear analysis techniques are used for analysis and design.

The block diagram in Fig. 2.1 shows the main components of a sinusoidal oscillator. It consists of an amplifier with a frequency–dependent forward loop gain G(jω) and frequency-selective network H(jω). In an actual oscillator circuit, normally no input signal will be present. We include an input signal here to explain the principle of operation. The output voltage is given by:

$$V_o = \frac{V_i G(j\omega)}{1+G(j\omega)H(j\omega)} \tag{2.1}$$

For an oscillator, the output V_o is nonzero even if the input signal V_i is a zero. At some frequency ω_0, the denominator will be zero. This leads to the well-known Nyquist criterion for oscillation

$$G(j\omega_0)H(j\omega_0) = -1$$

Those are the two conditions for oscillations to happen [9]:

i. The magnitude of the open loop transfer function should be unity

$$|G(j\omega_0)H(j\omega_0)| = 1$$

ii. And the phase shift is 180°. If a positive feedback is used, the loop phase shift must be 0°

$$argG\big((j\omega_0)H(j\omega_0)\big) = 180^{\circ}$$

Note that for the circuit to oscillate at one frequency, the oscillation criterion should be satisfied only at one frequency (i.e. ω_0); otherwise the resulting waveform will not be a simple sinusoidal.

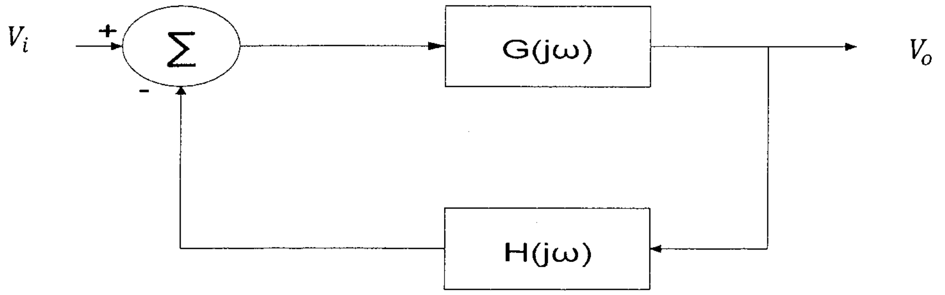


Fig.2. 1 A block diagram of an oscillator

There are various types of oscillators, such as LC-tuned oscillators and voltage-controlled oscillators (VCO). The LC-tuned oscillators consist of a transistor amplifier with three external impedances connected. Fig. 2.2 shows two commonly used configurations of LC-tuned oscillators. They are known as the Colpitts oscillator and the Hartley oscillator. Both utilize a parallel LC tank circuit connected between collector and base (or between drain and gate if a FET is used). In both circuits, the resistor R models the combination of the losses of the inductors, the load resistance of the oscillator and the output resistance of the transistor. If the frequency of operation is not too high, we can neglect the transistor capacitances, the frequency will thus be determined by the resonance frequency of the *tank* circuit.

Thus for Colpitts oscillator we have:

$$\omega_0 = \frac{1}{\sqrt{L\left(\frac{C_1 C_2}{C_1+C_2}\right)}}$$

For Hartley oscillator we have:

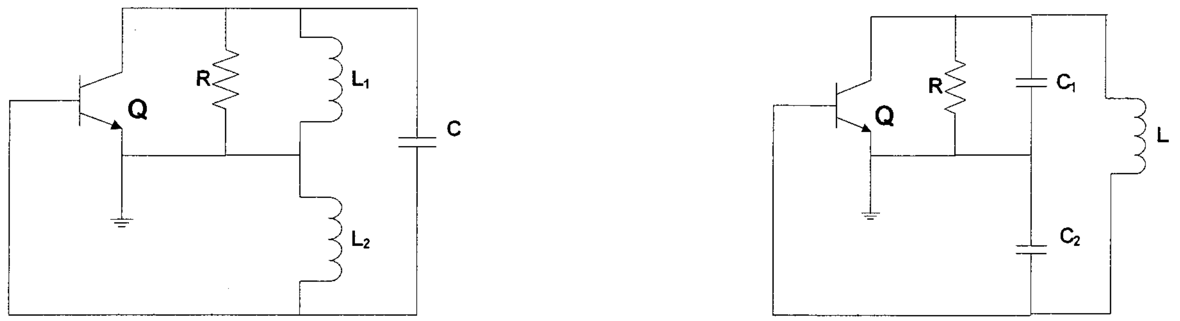
$$\omega_0 = \frac{1}{\sqrt{(L_1+L_2)C}}$$

A voltage Controlled Oscillator (VCO) is often designed using the Glapp-Gouriet configuration; where a varicap is a part of the LC-Tank circuit. One possible VCO configuration is shown in Fig. 2.3, with a varicap diode added in series between the C_2 and L, also for tuning, the varicap is connected to a DC voltage and C $_{is}$ a bypass capacitor.

The analytical frequency for this simple VCO is:

$$\omega_0 = \frac{1}{\sqrt{L\left(\frac{1}{C_1}+\frac{1}{C_2}+\frac{1}{C_D}\right)}} \quad ,$$

where C_D is the varicap junction capacitance



(a) (b)

Fig.2. 2 Two common configurations of LC- tuned oscillators: (a) Hartley (b) Colpitts

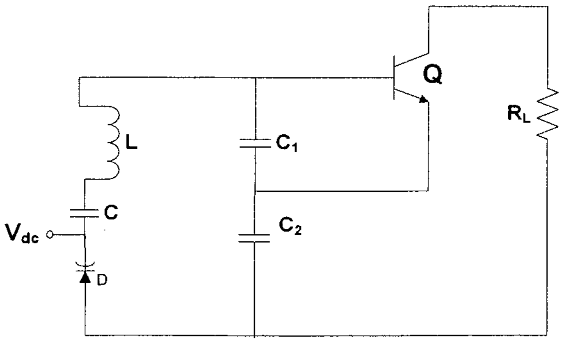


Fig.2. 3 Configuration of a VCO oscillator

2.3 Harmonic Balance (HB) Analysis

Harmonic balance (HB) is a nonlinear, frequency-domain, steady-state method. It is well-suited for simulating analog RF and microwave problems, since these are most naturally handled in the frequency-domain. The HB method is iterative. It is based on the assumption that for a given sinusoidal excitation there exists a steady-state solution that can be approximated to a satisfactory accuracy by means of a finite Fourier series. Consequently, the nonlinearities create discrete frequencies resulting in a spectrum of discrete frequencies at every node in the circuit. The currents flowing from nodes into linear elements, including all distributed elements, are calculated by means of a straightforward frequency- domain linear analysis. Currents from nodes into nonlinear elements are mostly calculated in the time-domain. Generalized Fourier analysis is used to transform from the time-domain to the frequency-domain. In some cases HB is more efficient than traditional time-domain since the number of Fourier coefficients might be smaller to achieve the same accuracy obtained by time-domain analysis [11]. Consider the DAE of a general nonlinear circuit:

$$\dot{q}(x(t)) + f(x(t)) = b(t), \tag{2.2}$$

where $b(t)$ is a vector of inputs, $f(.)$ is a vector function, $q(.)$ is a vector of nonlinear charges and $x(t)$ is a vector of unknowns (state variables of nodal voltages and currents). The key work of HB is to express the state variable x in Equation (2.2) as a truncated Fourier series:

$$x_j(t) = \mathcal{R}\left\{\sum_{i=0}^{i=a} X_j^i e^{ji\,\omega_0 t}\right\}, \tag{2.3}$$

where $x_j(t)$ is the j^{th} state variable in time domain, X_j^i is the Fourier coefficient of the i^{th} harmonic of the j^{th} state variable, ω_0 is the fundamental frequency, a is the number of harmonics and $\mathcal{R}$ is the real part. Let $\boldsymbol{X}, \boldsymbol{Q}, \boldsymbol{F}$ and $\boldsymbol{B}$ be vectors with all harmonics for all state variables in the frequency domain. $\boldsymbol{Q}$ and $\boldsymbol{F}$ can be evaluated using Discrete Fourier Transform (DFT):

$$\boldsymbol{Q}(\boldsymbol{X}) = \mathcal{F}\{\boldsymbol{q}(\mathcal{F}^{-1}(\boldsymbol{X})\},$$

$$\boldsymbol{F}(\boldsymbol{X}) = \mathcal{F}\{\boldsymbol{f}(\mathcal{F}^{-1}(\boldsymbol{X})\} \text{ ,}$$

where $\boldsymbol{q}$ is a vector of the nonlinear charges for all state variables in time-domain, $\boldsymbol{f}$ is a vector of the nonlinear functions of all state variables in time-domain and $\mathcal{F}$, and $\mathcal{F}^{-1}$ are the DFT and its inverse respectively. Define a diagonal matrix $\boldsymbol{\Omega}$:

$$\boldsymbol{\Omega} = \begin{bmatrix} \Omega & 0 & \cdots & 0 \\ 0 & \Omega & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Omega \end{bmatrix} , \qquad \text{where } \Omega = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & \omega_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a\omega_0 \end{bmatrix}$$

Then Equation (2.2) can be formulated in frequency-domain:

$$j\boldsymbol{\Omega}\, \boldsymbol{Q}(\boldsymbol{X}) + \boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{B} \text{ ,} \tag{2.4}$$

where j is the imaginary unit. The dimension of Equation (2.4) is equal to the number of state variables times the number of harmonics (a +1). The resulting algebraic equations of the nonlinear system in Equation (2.4) are solved with numerical methods like Newton-Raphson method (See Appendix A for Newton-Raphson method). Note that larger number of harmonics a improves the accuracy but slows the computational speed.

2.4 Multidimensional Methods

In radio frequency applications, electronic circuits include oscillatory signals with widely separated time scales. Due to this, numerical simulation becomes extremely time consuming, since the fastest time rate restricts the integration step size while the slowest time constant determines the interval of computation . Methods based on a multidimensional signal model [1,7] are an alternative approach. The multidimensional model decouples the widely separated rates into multiple time axes. Consequently, the differential algebraic equations (DAEs) are transformed into partial differential equations (PDEs). Numerical methods can then be used to solve the PDEs. This strategy causes savings in computation time and memory, if the arising partial differential algebraic equation (PDAE) can be solved efficiently.

The multidimensional idea was firstly introduced a few decades ago by Ngoyal Larchev, but only has been applied recently. Roychowdhury [1,7] applied multidimensional technique effectively to simulate circuits with widely separated times.

In this section, we will review multidimensional techniques; Multi-Time Partial Differential Equation (MPDE) and Warped Multi-Time Partial Differential Equation (WaMPDE). We will represent amplitude modulation (AM) using MPDE and frequency modulation (FM) will be represented using WaMPDE.

2.4.1 Multi-Time Partial Differential Equation (MPDE)

MPDE uses at least two time axes to represent signals with widely separated rates. Consider the general nonlinear circuit in Equation (2.2) and let $b(t)$ be a multi-rate input, with widely separated rates. Equation (2.5) is the equivalent MPDE of the circuit:

$$\left(\frac{\partial}{\partial t_1} + \frac{\partial}{\partial t_2}\right) q(\hat{x}) + f(\hat{x}) = \hat{b}(t_1, t_2)\,, \tag{2.5}$$

where $\hat{b}(t_1, t_2)$ is the multivariate form of inputs and $\hat{x}(t_1, t_2)$ is the multivariate form of unknowns vector (voltages and currents).

The state-space ordinary differential equations in Equation (2.2) are replaced by state-space multi-time partial differential equations in Equation (2.5). The circuit behaviour is separated in two different time axes; (t_2) and (t_1), one for slow variations (envelope) and the other for fast variations (oscillations). Note here the MPDE is in two dimensions (t_2, t_1), however multiple dimensions can be used depending on the number of signals in a system. In this work we will need only two dimensions (oscillations and envelope). The necessary condition for MPDE representation is the system in Equation (2.5) must be at least periodic in the fast time [2]. Fig. 2.4 shows how MPDE can be integrated in the $[t_2 - t_1]$ plane. Starting from appropriate initial conditions along t_1 and t_2 =0, the system is solved for each time step (h_2) along the slow time (t_2). The discretization along t_2 can be achieved, for example using Finite Difference (see Appendix B for Finite Difference) or Harmonic Balance.

The set of the discretized equations is solved with any numerical method for nonlinear equations, such as Newton-Raphson method. Note the initial condition ($t_2 = 0$) for MPDE is a set of discrete values along t_1.

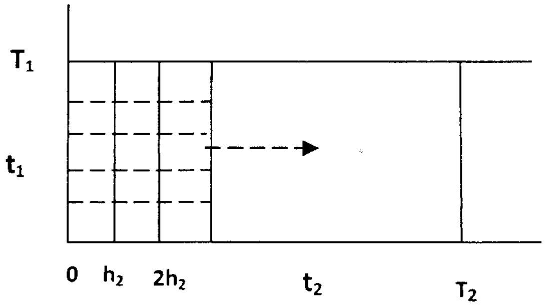


Fig.2. 4 Example showing how the integration in MPDE can be performed

Let's consider a simple oscillating signal with amplitude controlled by a slow sinusoidal signal [1].

$$y(t) = \sin\left(\frac{2\pi}{T_1}t\right)\sin\left(\frac{2\pi}{T_2}t\right), \qquad T_1 = 0.02\text{s}, \qquad T_2 = 1\text{s}, \tag{2.6}$$

The two tones at frequencies: $f_1 = \frac{1}{T_1}$ = 50 Hz and $f_2 = \frac{1}{T_2}$ = 1 Hz. There are 50 fast-varying sinusoids of period T_1= 0.02 s modulated by slow –varying sinusoids T_2= 1s. This signal is shown in Fig. 2.5. When $T_2 >> T_1$, consequently, many time steps are required to resolve all oscillations of the short period (T_1) during one long period (T_2). The same signal can be described using a multidimensional representation, where each time scale is described by its own variable. Accordingly, we obtain the function:

$$\hat{y}(t_1, t_2) = \sin\left(\frac{2\pi}{T_1}t_1\right)\sin\left(\frac{2\pi}{T_2}t_2\right) \tag{2.7}$$

By this transformation, we separate the envelope (slow) of the signal and its oscillation (fast) into two different time axes as shown in Equation (2.7). The transformed signal is shown in Fig. 2.6. It is obvious that the signal $\hat{y}(t_1, t_2)$ does not have as many undulations as the original signal $y(t)$. Also the transformed signal require fewer points which do not depend on the relative values of the periods T_1 and T_2. For example, Fig. 2.5 requires 20 points for each sinusoidal, hence a total number of 1000 points are needed to generate the signal. However, Fig. 2.6 was plotted with only 400 points on a uniform grid of 20×20, less than half of the points in Fig. 2.5. Such saving is more evident when the rates are widely separated. We can always recover the original signal by setting $t_1 = t_2 = t$.

MPDE is very useful for amplitude modulation, especially when the period separation factor m (m $= \frac{T_2}{T_1}$) is large. In many electronic circuits a separation factor of 1000 or more is common. The key idea of MPDE is the fact that the signal can be represented by fewer points than the original signal and the multidimensional form contains all the information needed to recover the original signal completely. MPDE doesn't however handle frequency modulation very well. In the next section we will present a generalized form of multi dimensional representation (WaMPDE) which deals well with FM problems.

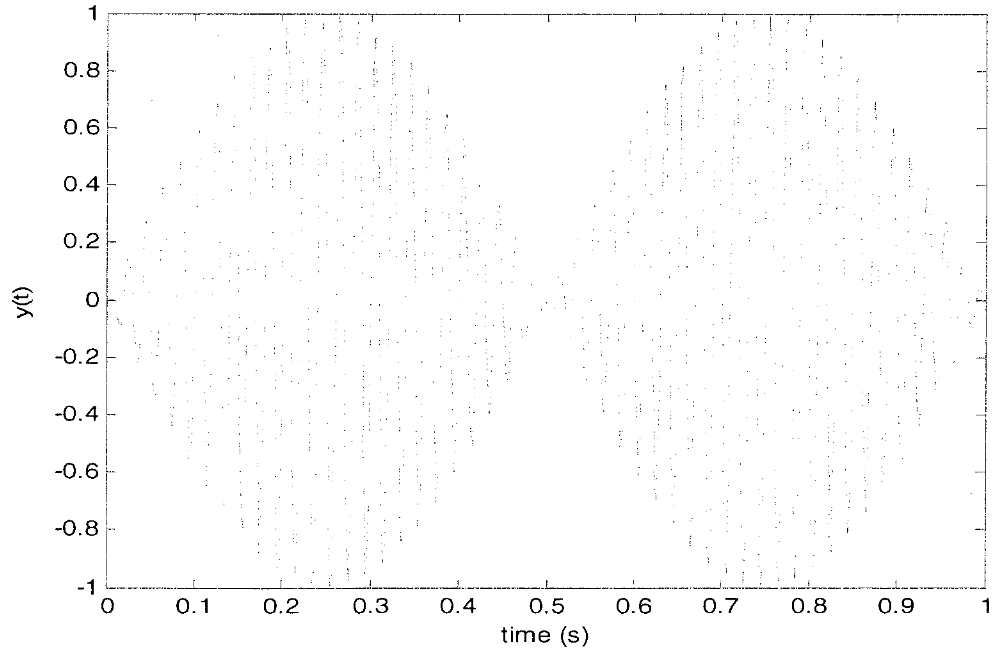


Fig.2. 5 Example of an AM signal y(t)

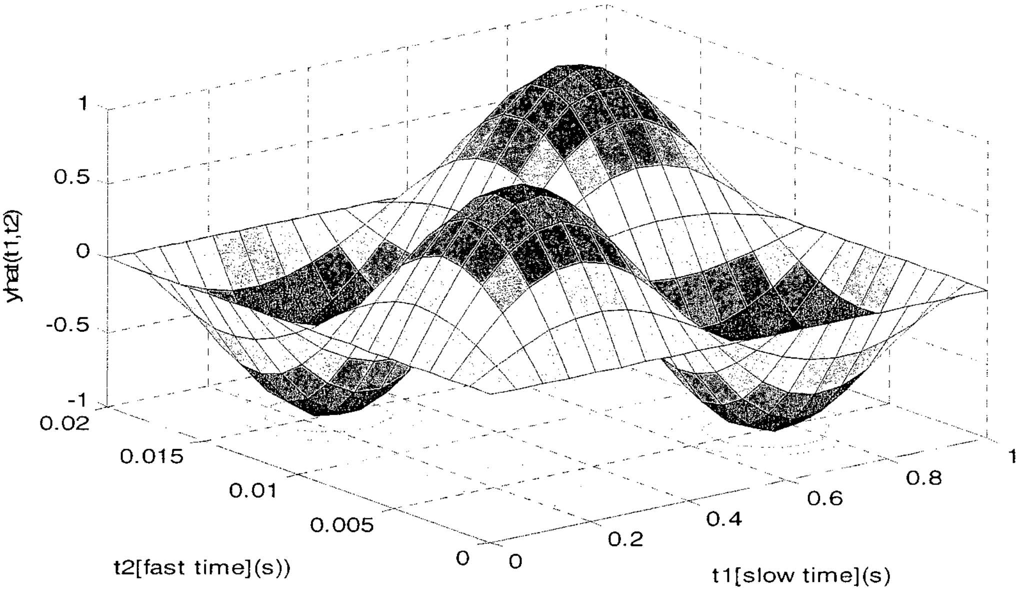


Fig.2. 6 MPDE representation of the AM signal y(t)

2.4.2 Warped Multi-time Partial Differential (WaMPDE)

Frequency modulation (FM) signals cannot, in general, be represented efficiently by the MPDE. To see this inefficiency, consider the following prototypical FM signal:

$$x(t) = \cos\big(\varphi(t)\big) = \cos\big((2\pi f_0\, t) + \; k\cos(2\pi f_2\, t)\big) \qquad (2.8)$$

The local frequency or named instantaneous frequency is modulated by a slow sinusoidal signal:

$$f(t) = \frac{1}{2\pi} \times \frac{\partial\varphi(t)}{\partial t} \;=\; f_0 \;\text{-}\, \mathrm{k}\, f_2 \;\sin(2\pi f_2\, t)\,, \qquad (2.9)$$

with f_0 =1 MHz, f_2 =20 KHz and k= 8π. The waveform of this signal is shown in Fig. 2.7. It's obvious the waveform is unclear and difficult to analyze. Now, let's apply MPDE with two time scales to this signal. Following the previous procedure, we will define the multidimensional signal to be:

$$\hat{x}\,(t_1\,,t_2)\;=\cos\big((2\pi f_0\,t_1\,)+\;k\cos(2\pi f_2\,t_2\,)\big) \qquad (2.10)$$

The MPDE representation is shown in Fig. 2.8, here we have the fast time t_1 representing the oscillation and the slow time t_2 representing the modulation. The bivariate waveform of this signal $\hat{x}(t_1\,,t_2)$ is periodic in t_1 and t_2, however the period along t_1 is not constant with t_2. Unfortunately $\hat{x}(t_1\,,t_2)$ is not as clear as been observed before in the case of AM. When k >> 2π , i.e, k ≈ $2\pi m$ for some large integer m, then $\hat{x}(t_1\,,t_2)$ will have about m oscillations as a function of t_2 over one period of the slow variations, which will be difficult to sample along t_2. Therefore in general MPDE is inefficient to represent FM signals.

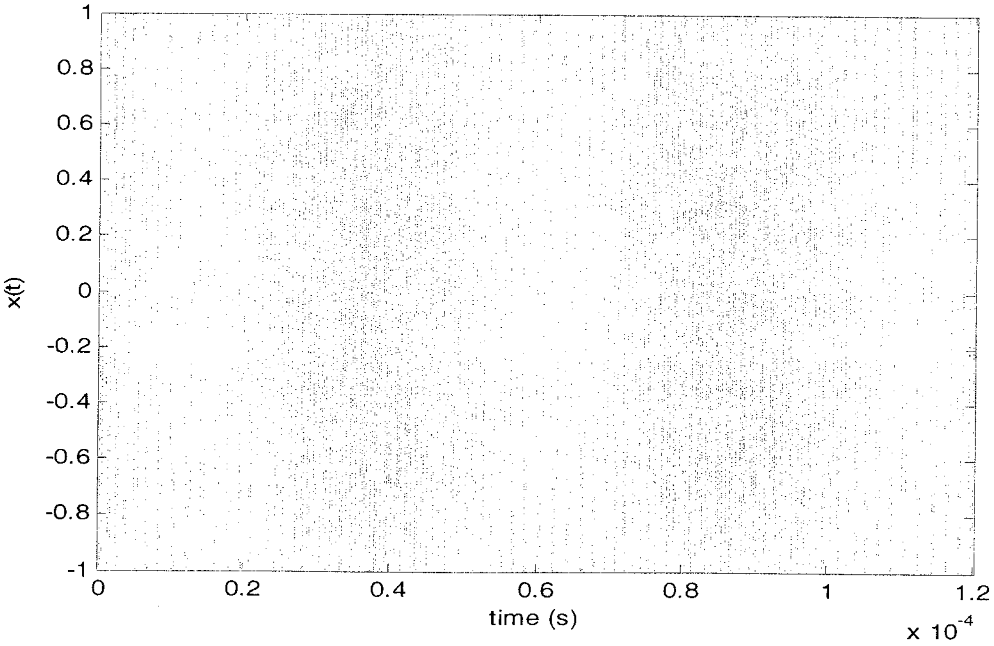


Fig.2. 7 Example of FM signal

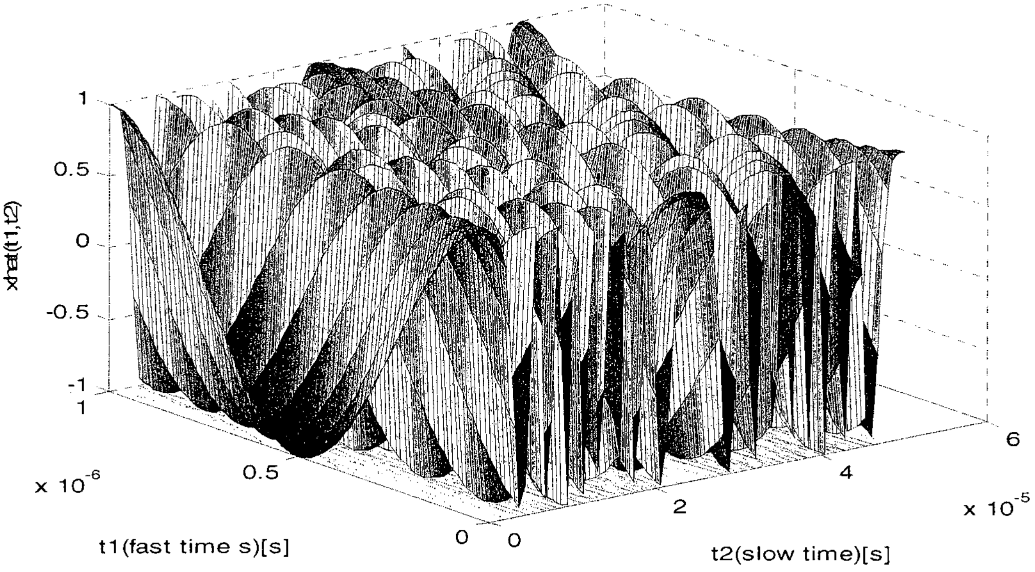


Fig.2. 8 FM signal represented by MPDE

Narayan and Roychowdhury [1] introduced a version of multidimensional representation named Warped Multi-time Partial Differential Equation (WaMPDE) to represent FM signals. WaMPDE makes the waveform uniform by warping at least one time scale of the MPDE, this effectively results in stretching and squeezing the t_1 time axis at different times to even out the period of the fast undulations. Equation (2.11) is the WaMPDE representation of the system in Equation (2.2).

$$\omega(\tau_2)\frac{\partial}{\partial\tau_1}q(\hat{x}) \;+\; \frac{\partial}{\partial\tau_2}q(\hat{x})+f(\hat{x})=\hat{b}(\tau_1,\tau_2)\,, \tag{2.11}$$

where τ_1 is the warped time scale, τ_2 is the unwarped time and $\omega(\tau_2)$ is the warping function. Note that τ_2 is equal to the real time t, and τ_1 is no longer equal to the real time t, we refer to τ_1 as the warped time. The relation between τ_1 and τ_2 is given by:

$$\tau_1=\int_0^t \omega(\tau_2)d\,\tau_2$$

The warping function depends on the unwarped time variable. Note that the warping function is an extra unknown and it requires an additional equation; otherwise the WaMPDE has more unknowns than equations. Thus we need to balance the number of unknowns with the number of equations. Balancing the equations is usually accomplished by not specifying the warping function *a priori*, but instead a phase condition is imposed on the WaMPDE function. The phase condition requires that the phase of the function along t_1 should vary slowly or not all as t_2 changes [1]. When Equation (2.11) is solved in the frequency-domain, it is a common practice to impose the phase condition by forcing a particular phase in one of the state variables, *e.g.* the imaginary part to zero. This is equivalent to selecting one of the infinite solutions that only differ in phase. In time-domain, imposing the phase condition can be done by setting one sample of the derivative of one of the state variables equal to zero.

We apply WaMPDE to the FM tone in Equation (2.8), we use τ_1 instead of t_1 to normalize the local frequency and τ_2 is the same as the real time (t_2) axis:

$$\hat{x}_2(\tau_1, \tau_2) = \cos(\tau_1)\,, \tag{2.12}$$

with $x(t) = \hat{x}_2(\phi(t), t)$. The warped time is a function of time

$$\tau_1 = \phi(\tau_2) = 2\pi f_0 \tau_2 + k\cos(2\pi f_2\, \tau_2) \tag{2.13}$$

The result of the two dimensional representation is shown in Fig. 2.9. In this figure we can notice there are fewer undulations than the MPDE representation in Fig. 2.8 and thus the signal can be sampled with relatively fewer points since the rescaled time axis stretches and squeezes the real time axis by different amounts at different times to even out the period of the fast oscillations. The mapping between warped time and real time is shown in Fig. 2.10.

It's worth noting that the solution of WaMPDE is not unique, WaMPDE is related to the choice of warped time τ_1. We can recover the original signal by setting:

$$\tau_2 = t \quad \text{and}$$

$$\tau_1 = \phi(\tau_2) = \phi(t)$$

It's clear that WaMPDE can handle FM signals well when the rates of frequencies are widely separated. Also MPDE can be viewed as nothing but special case of WaMPDE; we have MPDE, when the warped time is set to the real time.

We need to specify the boundary conditions (BCs) for the differential equations of the WaMPDE. One shortcoming of WaMPDE is that it is sensitive to BCs, since BCs need to be specified along the whole vertical line $x(\tau_1, 0)$, as shown in Fig. 2.11. A simpler approach to approximate the BCs [6] of WaMPDE is to find the initial value $x(0)$ of the ODEs of the system. This value will correspond to $\hat{x}(0,0)$ in WaMPDE, and then we use one full period or the interpolation of two periods of the ODE solution as a BC for $(\tau_1, 0)$ line. In addition, a periodic boundary condition is imposed:

$$\hat{x}(0, \tau_2) = \hat{x}(2\pi, \tau_2)$$

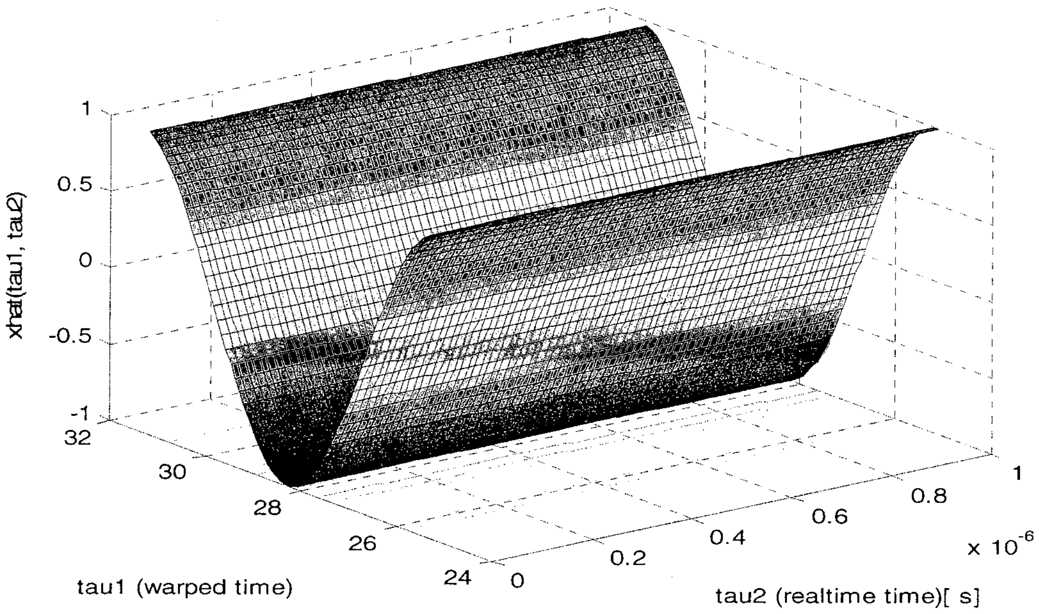


Fig.2. 9 FM signal represented by WaMPDE

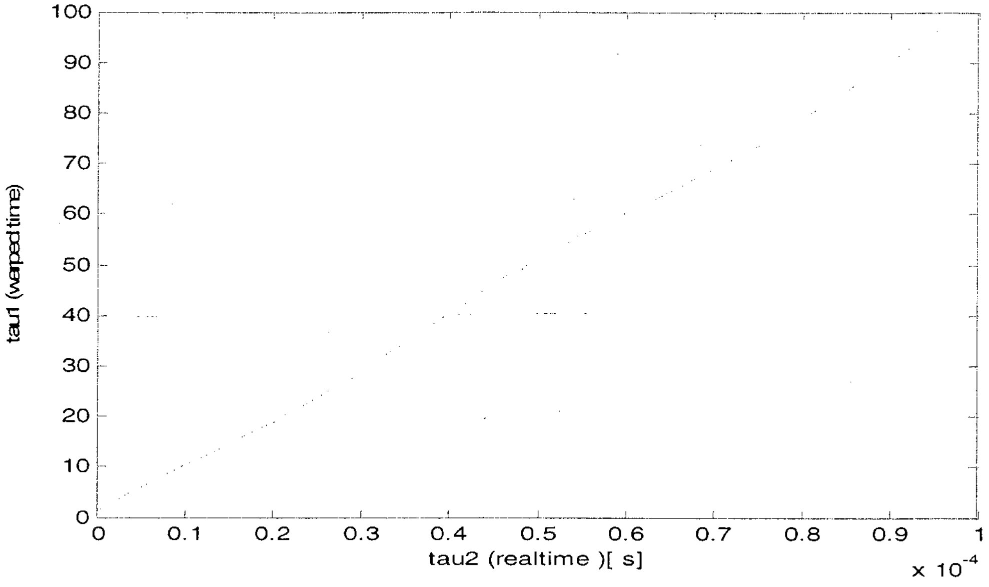


Fig.2. 10 The relation between tau1 (τ_1) and tau2 (τ_2)

The resulting solution has the fast oscillation along τ_1 time axis and the envelope along τ_2 time axis. A much improved method to approximate the BCs is found in [6].

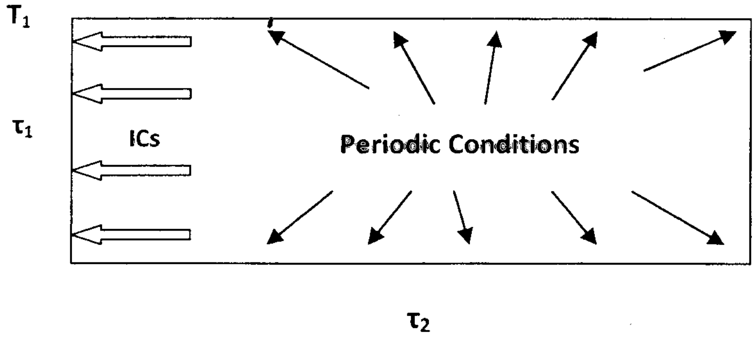


Fig.2. 11 Graphical representation shows boundary conditions

Solving a WaMPDE system using HB:

HB method can be used to solve the WaMPDE along the warped time (τ_1), the solution along the warped time axis (τ_1) is solved in frequency or time-domain. The unknowns are the

Fourier coefficients along τ_1 corresponding to each value of the real time τ_2. In HB the state variable x is expressed as a truncated Fourier series:

$$x(\tau_1, \tau_2) = \mathcal{R}\left\{\sum_{i=0}^{i=a} X^i(\tau_2)e^{ji\,\tau_1}\right\}, \tag{2.14}$$

The period in the warped time (τ_1) is normalized to 2π (i.e. ω=1).

The Equation (2.2) is formulated with Fourier transform:

$$ji\omega_0(\tau_2)Q^i + \frac{dQ^i\,(\tau_2)}{d\tau_2} + F^i(\tau_2) = B^i(\tau_2) \tag{2.15}$$

The fundamental frequency ω_0 is an extra unknown. So a phase condition is usually imposed as we mentioned earlier by forcing the imaginary part of one of the state variables to be constant. For example, we can set the imaginary part of the fundamental frequency to a zero:

$$\Im(\mathrm{X}^1) = 0 \tag{2.16}$$

Note Equation (2.15) and Equation (2.16) form a + 1 equations in the same number of unknown functions of τ_2. The resulting balanced equations are solved with Newton-Raphson method.

2.5 Formulation of WaMPDE on oscillators

In this section we will show how to formulate the WaMPDE method to analyze a simple voltage controlled oscillator [2], in which the system ODE is replaced with partial differential equations. The periodic solution along the warped time is solved with Finite Difference Time Domain (FDTD) or HB (Frequency-domain). The solution along the unwarped (real) time is solved using time marching method (time-domain).

The circuit in Fig. 2.12 shows the schematic of the VCO. The VCO consists of an LC tank in parallel with a nonlinear resistor. The capacitance is varied with a separate control voltage.

The values of the elements in Fig. 2.12 are: $C = 1/(2\pi)\mu\text{F}$, $C_m = 1/(4\pi)\mu\text{F}$ and $L = 1/(2\pi)$ μH.

The nonlinear resistor is given by:

$$i_R = f(u) = (G_0 - G_\infty)\tanh(u) + G_\infty u\,, \tag{2.17}$$

with $G_\infty = 0.25$ and $G_0 = -0.1$

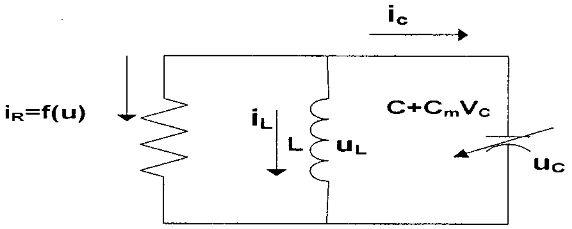


Fig.2. 12 A simple ideal VCO oscillator

Here the fast oscillation and the amplitude of the capacitor voltage u_c and inductor current i_L are modulated with a much slower rate. We will find the transient response of this system with the ODE method and the WaMPDE method. We will compare the solutions of the two methods for two different variations of the control voltage.

2.5.1 ODE Analysis

The control voltage V_c:

$$V_c = 1.5cos\left(\frac{2\pi}{T}\right)t\,, \tag{2.18}$$

is varied with a fast oscillation $T_1 = 30\ \mu s$ and a slow oscillation $T_2 = 1\,ms$, the slow oscillation V_c is shown in Fig. 2.13.

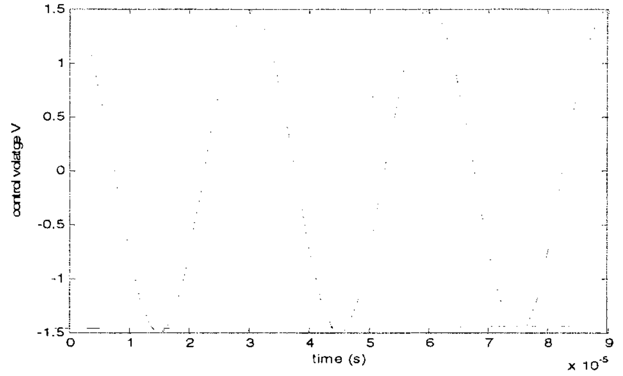


Fig.2. 13 Control voltage –slow

The initial local frequency (i.e. *t*=0) is:

$$f_0 = \frac{1}{2\pi\sqrt{L(C+C_m V_{c0})}} = 0.75 \text{ MHz} \tag{2.19}$$

The local frequency as a function of time is given by:

$$f_L = \frac{1}{2\pi\sqrt{L(C+C_m V_c)}} = \frac{1}{\sqrt{1+0.75\cos{(\omega_c t)}}} \tag{2.20}$$

Fig. 2.14 and Fig. 2.15 show the waveforms of the local frequency under the two variations of control voltage V$_{c.}$

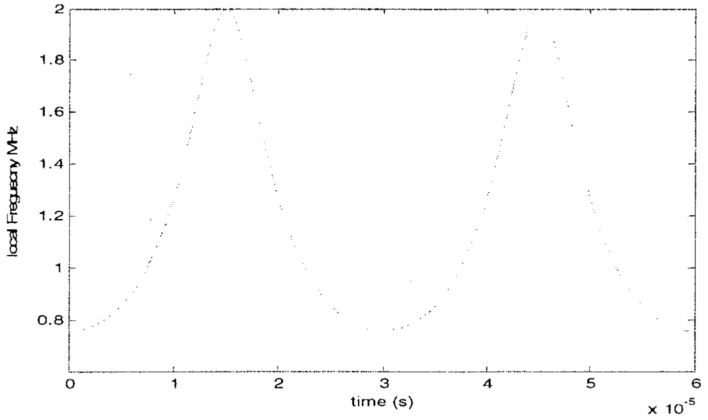


Fig.2. 14 Local frequency under fast control voltage

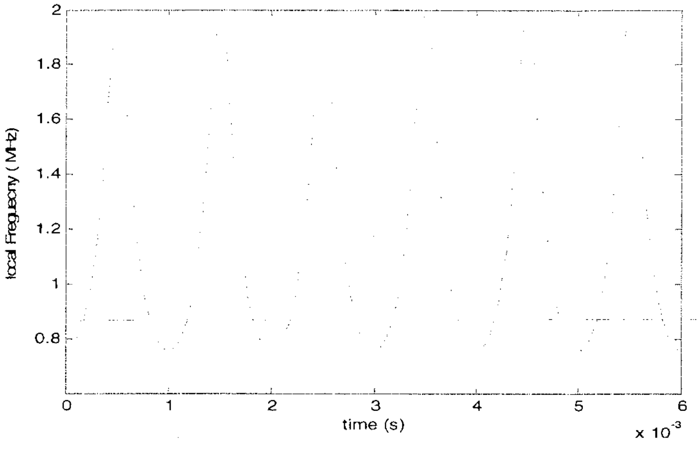


Fig.2. 15 Local frequency under slow control voltage

Using KCL, the oscillator currents satisfy:

$$i_R + i_L + i_C = 0\ , \tag{2.21}$$

with

$$\begin{cases} i_L = \dfrac{1}{L}\displaystyle\int u_L\, dt \\ i_C = \dfrac{dQ}{dt} = \dfrac{d(\,C + C_m V_c)\ u_c}{dt} = (C + C_m V_c)\dfrac{du_c}{dt} + C_m \dfrac{dV_c}{dt} u_c = 0 \end{cases}$$

Using KVL, the oscillator voltages satisfy:

$$u_L = u_c = u_R \tag{2.22}$$

The ordinary differential equations governing the oscillator can be written as:

$$\begin{cases} u_c - L\dfrac{di_L}{dt} = 0 \\ i_L + (C + C_m V_c)\dfrac{du_c}{dt} + C_m \dfrac{dV_c}{dt} u_c + f(u) = 0 \end{cases} \tag{2.23}$$

The capacitor voltage u_c and the inductor current i_L are the state variables. Their initial values are set to be 0 V and -1 A. Using the ode45 function in Matlab, we obtain the waveforms of capacitor voltage under the fast variation as shown in Fig. 2.16. The simulation is run again under slow variation as shown in Fig. 2.17.

Obviously the control voltage modifies the local frequency; it also modifies the shape of the envelope. Simulation time for fast varying was about 2.2 seconds, however for the slow variation was about 200 seconds. Under the slow variation, the simulation took a long time. In this example, the effectiveness of the ODE depends on rate of the control voltage. In the next section, we will apply the WaMPDE method and observe its effectiveness.

2.5.2 WaMPDE Analysis

We will apply WaMPDE method to analyze the VCO. We mentioned in Section 2.4 that the relation between the warped time τ_1 and the real time τ_2 is given by:

$$\tau_1 = \int_0^t \omega(\tau_2) d\,\tau_2$$

To apply the WaMPDE method: Firstly, we set the initial conditions in the grid $(\tau_1, \tau_2\ )$. Secondly, we convert Equation (2.23) into multi-time differential equations as shown in Equation (2.24).

$$\begin{cases} F_1(\tau_1,\tau_2) = u_c - L\left(\omega(\tau_2)\dfrac{\partial i_L}{\partial \tau_1} + \dfrac{\partial i_L}{\partial \tau_2}\right) = 0 \\ F_2(\tau_1,\tau_2) = i_L + \left(C + C_m V_c(\tau_2)\right)\left(\omega(\tau_2)\dfrac{\partial u_c}{\partial \tau_1} + \dfrac{\partial u_c}{\partial \tau_2}\right) + C_m \dfrac{\partial V_c}{\partial \tau_2} u_c + f(u_c) = 0 \end{cases} \tag{2.24}$$

The warping function $\omega(\tau_2)$ is an added unknown that must be calculated for each time step along τ_2. Thus, we need to set the phase of one of the variables to an arbitrary value, we then have:

$$F_3(\tau_1,\tau_2) = u_c(0,\tau_2) = 0 \tag{2.25}$$

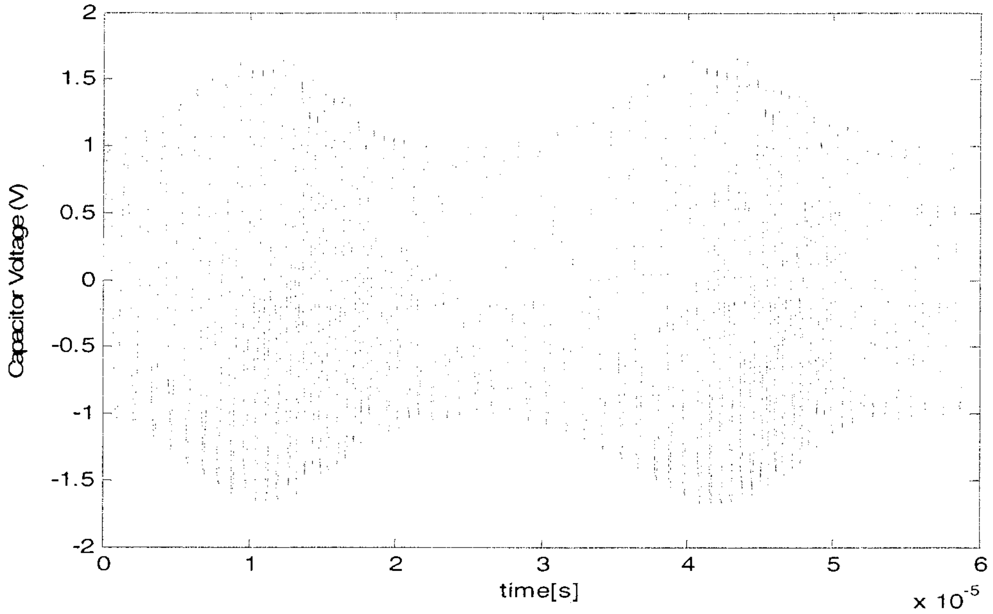


Fig.2. 16 Capacitor voltage under fast control voltage

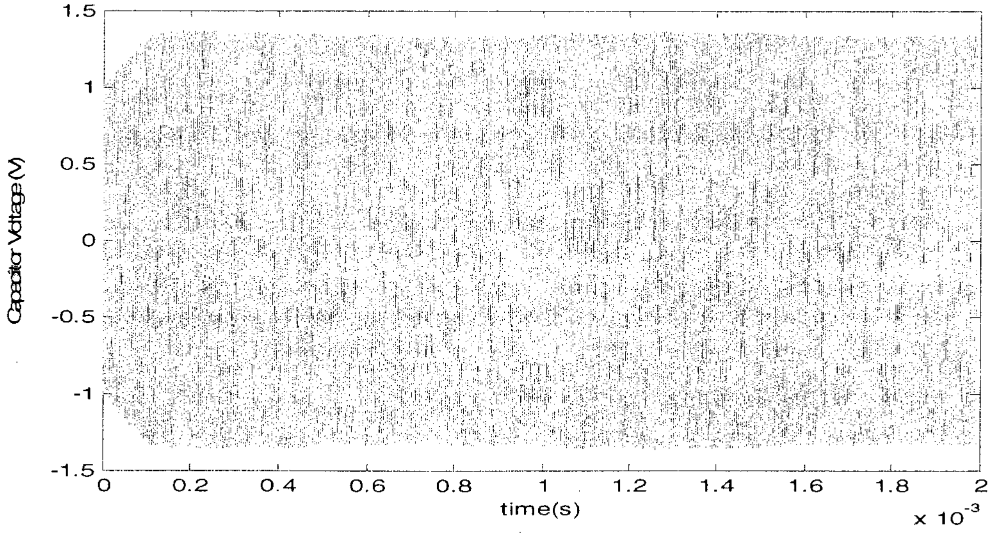


Fig.2. 17 Capacitor voltage under slow control voltage

Next we solve the vector of the nonlinear equations:

$$F = |F_1 \ F_2 \ \ F_3 \ |^T$$

The nonlinear equations are discretized with the five point formula along τ_1 and with Backward Euler Rule along τ_2 (see Appendix B for five point formula and Backward Euler Rule). Finally, the unknowns which are along τ_1 are solved with Newton-Raphson for each time step along τ_2.

In this VCO, we use the same initial guess values for voltage and current used in the ODE (0 V and -1 A). The boundary condition of the capacitor voltage is set to be a sinusoidal wave along τ_1 with amplitude 1 V. The number of samples used for the discretization along both time axes is n1 =n2 =20. The time step are h1 = 2*pi/n1, for τ_1 axis and h1=T/n2, for τ_2 axis.

The WaMPDE of the capacitor voltage under the fast varying control voltage is shown in Fig. 2.18. Like the ODE representation, WaMPDE represents frequency modulation and amplitude modulation, but the waveform is much clearer and easier to analyze. The local frequency associated with the circuit is shown in Fig. 2.19, which as expected, is identical to the local frequency shown in Fig. 2.14. The relation between warped time and real time are shown in Fig. 2.20. Here the warped time represents the phase shift of the carrier for each value of real time. The computational time was 2.4 seconds.

The WaMPDE simulation is run again under the slow varying control voltage, the result is shown in Fig. 2.21. As expected, the envelope changes very little for the capacitor voltage, which is consistent with ODE results, but again, we have a clearer waveform. The local frequency associated with the circuit is shown in Fig. 2.22, which as expected, is identical to the local frequency shown in Fig. 2.15. The relation between warped time and real time is shown in Fig. 2.23. The computational time is significantly reduced, while ODE needed 200 second, WaMPDE needed only 34 seconds.

These results show the effectiveness of the WaMPDE method in terms of saving the computational effort (i.e. time). There was no difference between the two methods under the fast varying voltage but there was a considerable saving by the WaMPDE method for the slow varying voltage.

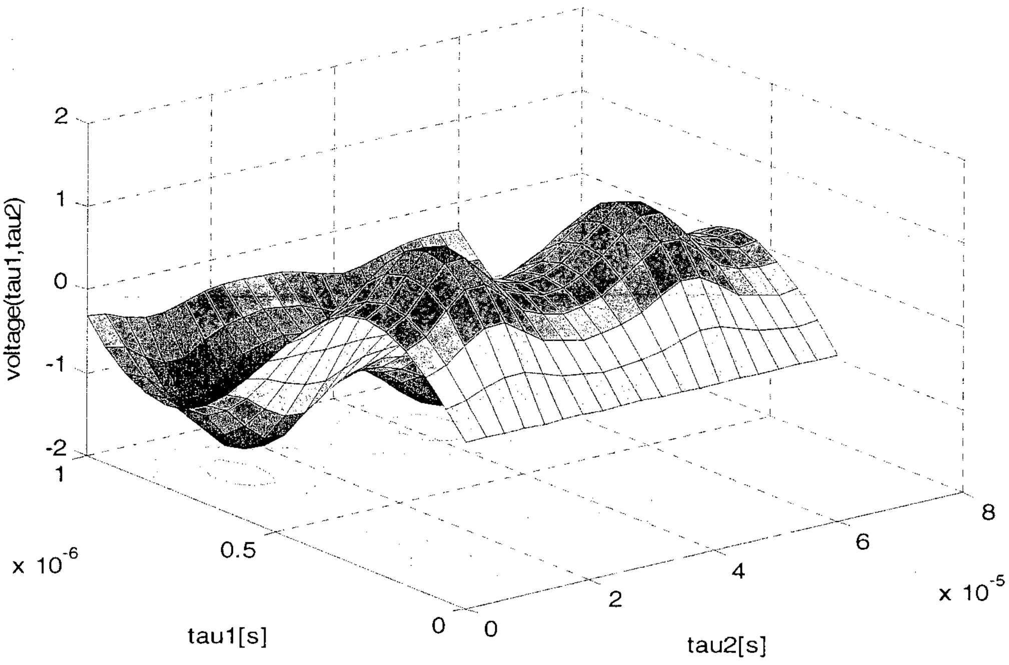


Fig.2. 18 Capacitor voltage under fast control voltage represented by WaMPDE

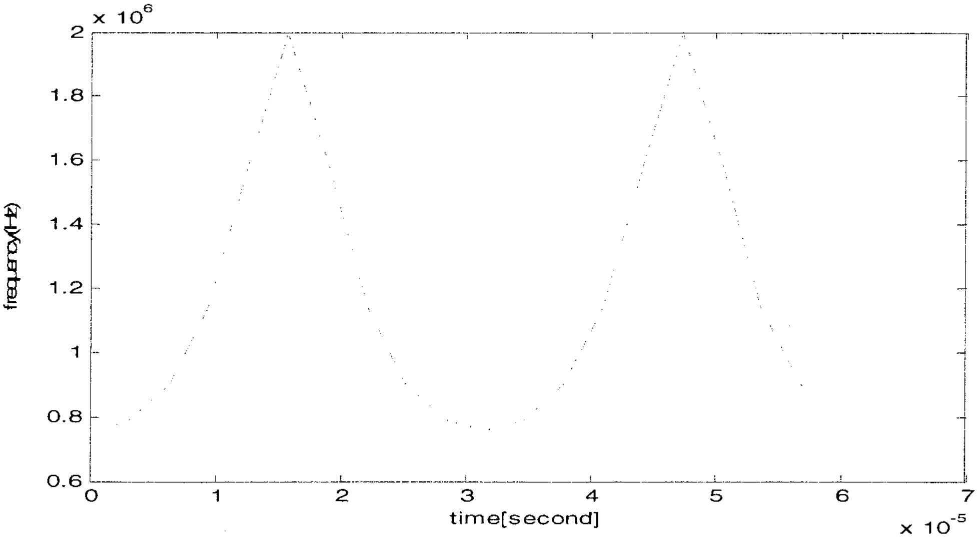


Fig.2. 19 Local frequency under fast control voltage

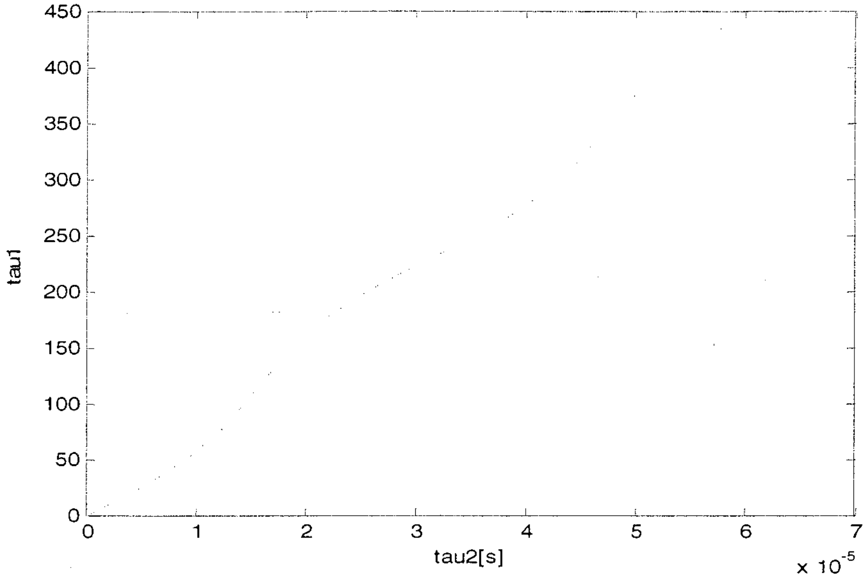


Fig.2. 20 Relation between warped time and real time (fast variation)

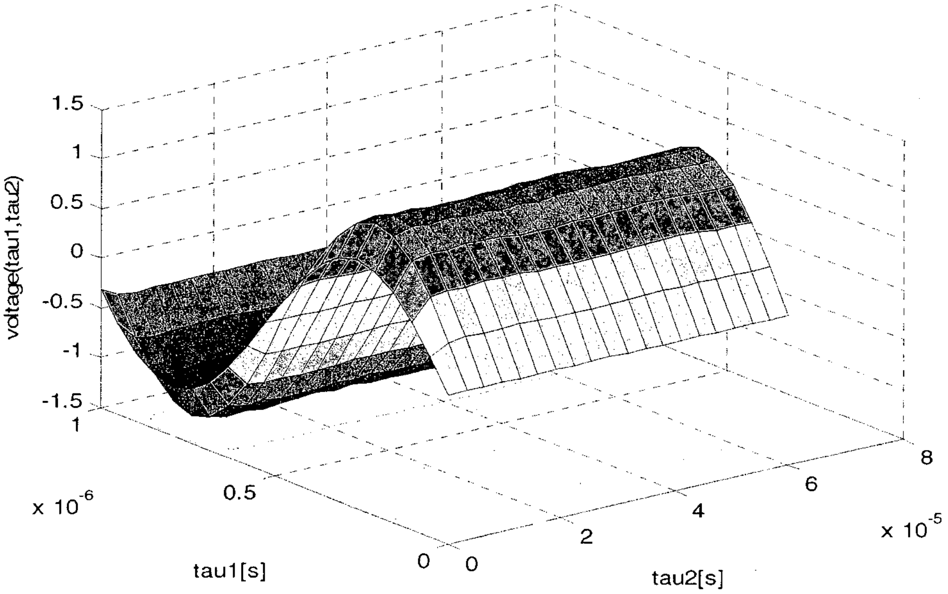


Fig.2. 21 Capacitor voltage under slow control voltage represented by WaMPDE

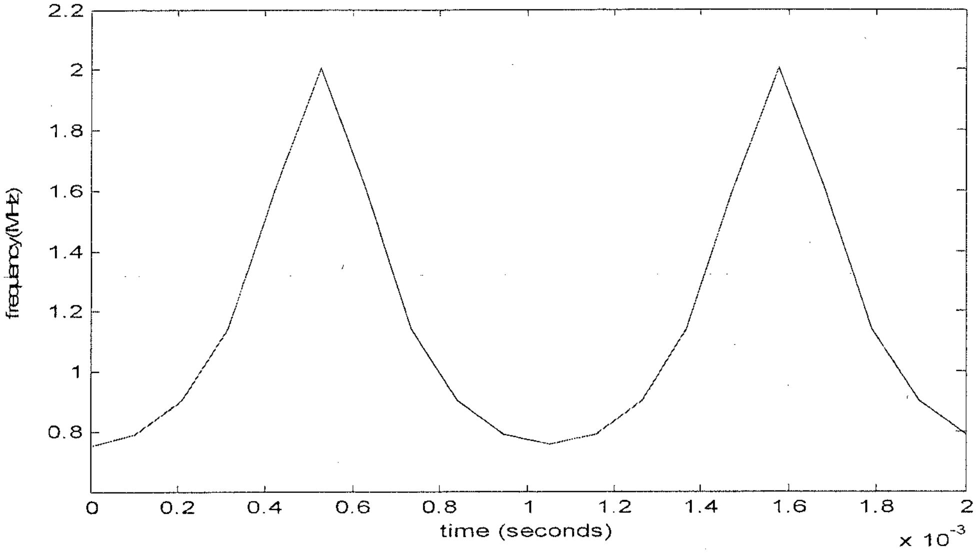


Fig.2. 22 Local frequency under slow control voltage

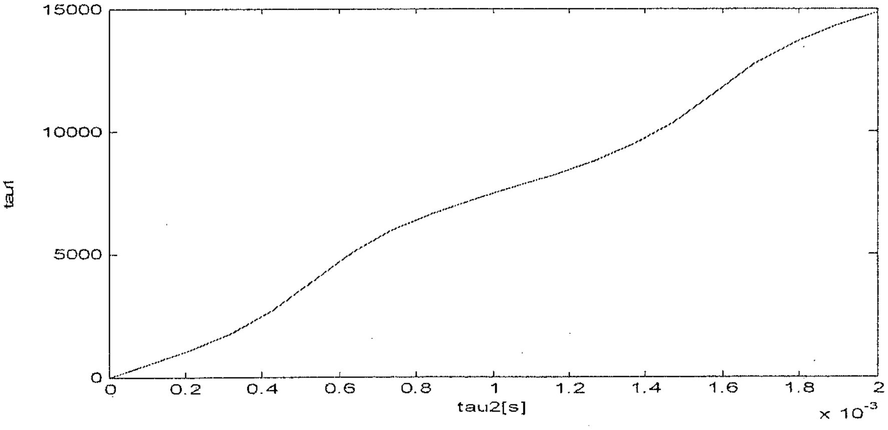


Fig.2. 23 Relation between warped time and real time (slow variation)

2.6 Steady-State Analysis of Oscillators

In this section a review explains how the method in Ref. [6] captures the steady-state of oscillators. For the steady-state analysis, the method uses WaMPDE to find a good initial guess for HB in a fast and a robust way. The general nonlinear system in Equation (2.2) can be written by separating the linear and nonlinear parts as follows:

$$Gx(t) + C\frac{dx(t)}{dt} + \frac{dQ(x(t))}{dt} + I(x(t)) = S(t) \ , \qquad (2.26)$$

where *x(t)* is vector of state variables(the nodal voltages vector), *G* is conductance matrix, *C* is matrix of linear capacitances, $I(x(t))$, and *Q*(x(*t*)) are nonlinear functions representing nonlinear devices and *S(t)* is the source vector.

The Equation (2.26) is transformed into the WaMPDE from:

$$(G + C\mathbf{j}i\omega_0)X^i + C\frac{dX^i}{d\tau_2} + \mathbf{j}i\omega_0 Q^i + \frac{d\,Q^i}{d\,\tau_2} + I^i - S^i = 0 \qquad (2.27)$$

and the phase condition in Equation (2.16)

$$\Im(\mathrm{X}^1) = 0$$

where i is the harmonic number and **j** is the imaginary unit. The WaMPDE solves Equation (2.27) and Equation (2.16) along the warped time for each time step along the real time τ_2. The solution samples are along the warped time.

The method implementation is accomplished in two steps:

I) In the first step, the multi-dimensional simulation runs until the initial transient starts to disappear. This step is accomplished in the minimum number of Newton iterations possible. At this point, the solution is assumed to be close to steady-state in which the local frequency function has become constant, and the difference between two successive periodic solutions (error) along the warped-time (τ_1) has become smaller than some desired tolerance ϵ_{ds}. The transient envelope is stopped and the second step begins.

II) In the second step, the regular HB starts, here the solution is switched from multi-time to single time. Note the switching is achieved by setting all the derivatives in Equation (2.27) with respect to τ_2 to zero. The regular HB analysis is started with initial conditions set to the last multi-dimensional solution. Since this initial condition is close to the steady-state, only few Newton iterations are needed for the regular HB.

A flow chart in Fig. 2.24 shows the method algorithm. The simulation is started with initial conditions set to the DC bias point. An excitation current:

$$i_s(\tau_1, \tau_2) = \begin{cases} I_0 \cos(\tau_1) & \text{if } \tau_2 \le t_i \\ 0 & otherwise \end{cases},$$

is injected from the ground into one of the nodes where we expect oscillations. The excitation current is very small (I_0 is about a few µA), and t_i is the size of the first time- step in the direction of τ_2. The excitation current is needed to start the oscillation by moving the system away from the unstable DC equilibrium point.

The method employed several provisions to improve the efficiency of the simulations:

1. For the integration along the real time, Backward Euler Rule is used. It allows numerical damping [7]. The advantage of this is that if the oscillations along τ_2 are damped, this will allow to use longer time step.
2. To further reduce the number of iterations to get close to the steady-state, the tolerance of the Newton method is adaptively controlled during the transient evolution (first step). Since the interest in the steady-state solution, the tolerance of Newton method needs not to be very small during the transient evolution. This provision will reduce the number of Newton iterations for each solution of τ_2. The Newton tolerance (*Tol)* is set to:

$$Tol = \max\{0.1\% X^i_{max}, 10^{-7}\},$$

where X_{max}^{i} is the amplitude of the largest oscillation in the circuit at the previous value of τ_2. When regular HB is started, for better accuracy, smaller tolerance of Newton Method is used (typically in the order of 10^{-9}). Smaller tolerance is used since the solution is very close to the steady-state.

3. Adaptively controlling the number of harmonics for each state variable x independently. At the start of the simulation (τ_2=0), only a small number of harmonics will be used since the oscillations started at the DC bias point. For each solution away from DC bias points along τ_2, the values of the last two harmonics are considered. If they are greater than some threshold values, then the number of harmonics for that state variable is increased by one, otherwise it is left unchanged for the next time step. The flow chart in Fig. 2.25 shows the implementation of adaptive harmonic balance. There $\boldsymbol{A}_j$ and $\boldsymbol{B}_j$ are the magnitude of the last two harmonics for the j^{th} state variable. The advantage of using adaptive harmonic balance is that the number of harmonics is only increased as needed.
4. The size of the time step (h_2) along τ_2 is varied according to the number of Newton iterations (n) required to achieve the Newton tolerance. The flow chart of adaptive time step is shown in Fig. 2.26. Here ***F*** is the time step factor, n_{min} and n_{max} are the decision factors, if the number of Newton iterations is larger than n_{max}, the time step is reduced, if it is smaller than n_{min}, the time step is enlarged, otherwise it is kept unchanged for the next calculation.

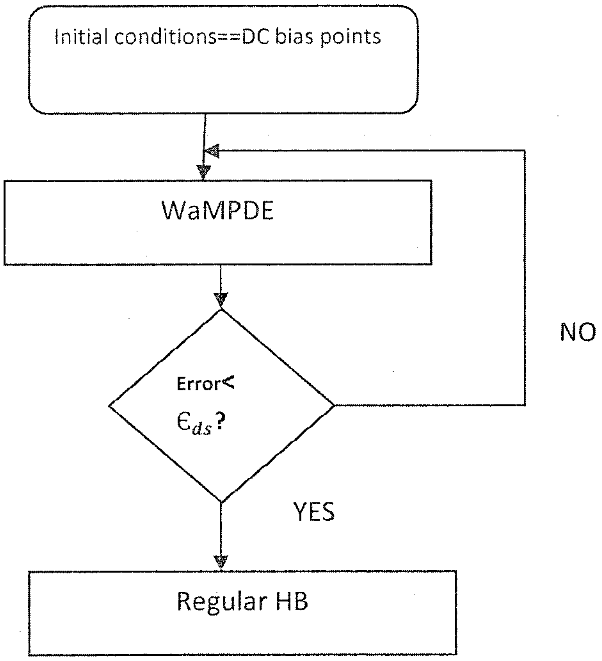


Fig.2. 24 Flow chart corresponding to the method in Ref [6]

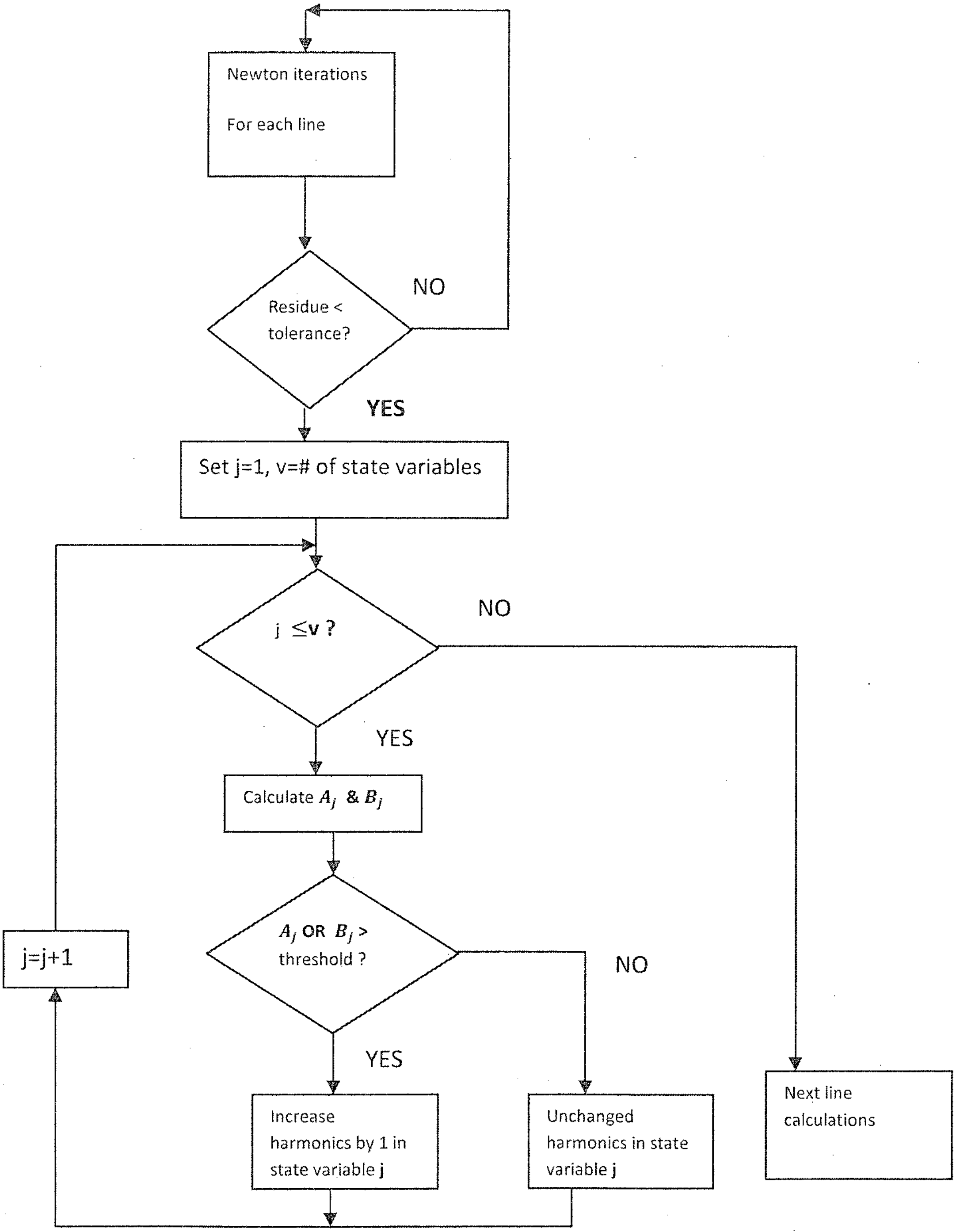

Newton iterations
For each line
Residue < tolerance?
NO
YES
Set j=1, v=# of state variables
j ≤v ?
NO
YES
Calculate A_j & B_j
A_j OR B_j > threshold ?
NO
YES
j=j+1
Increase harmonics by 1 in state variable j
Unchanged harmonics in state variable j
Next line calculations

Fig.2. 25 A flow chart of adaptive harmonic balance

As stated before, the solution is considered very close to steady-state when the local frequency function has become constant and the difference between two successive periodic solutions (error) along τ_2 becomes smaller than ϵ_{ds}. To measure the difference of periodic solutions, we calculate the maximum difference of voltages:

$$\max_i \left\{ \frac{\left| X_j^i(\tau_2 + h_2) - X_j^i(\tau_2) \right|}{\left| X_j^i(\tau_2 + h_2) \right|} \right\} < \epsilon_a \, , \tag{2.28}$$

where ϵ_a is relative tolerances, j=0,1,2....v,v is the number of nodal voltages, i =1,2,3... a, where a is the number of harmonics of the voltage j and X_j^i is the solution of i th harmonic of the j th nodal voltage. Equation (2.28) calculates the error for a nodal voltage for all harmonics and selects the maximum tolerance. The calculation is done for all the nodal voltages. And then select the maximum difference of the nodal voltages. If $X_i(\tau_2 + h_2)$ is close to zero we use the absolute tolerance ϵ_b:

$$\max_i \left| X_j^i(\tau_2 + h_2) - X_j^i(\tau_2) \right| < \epsilon_b \tag{2.29}$$

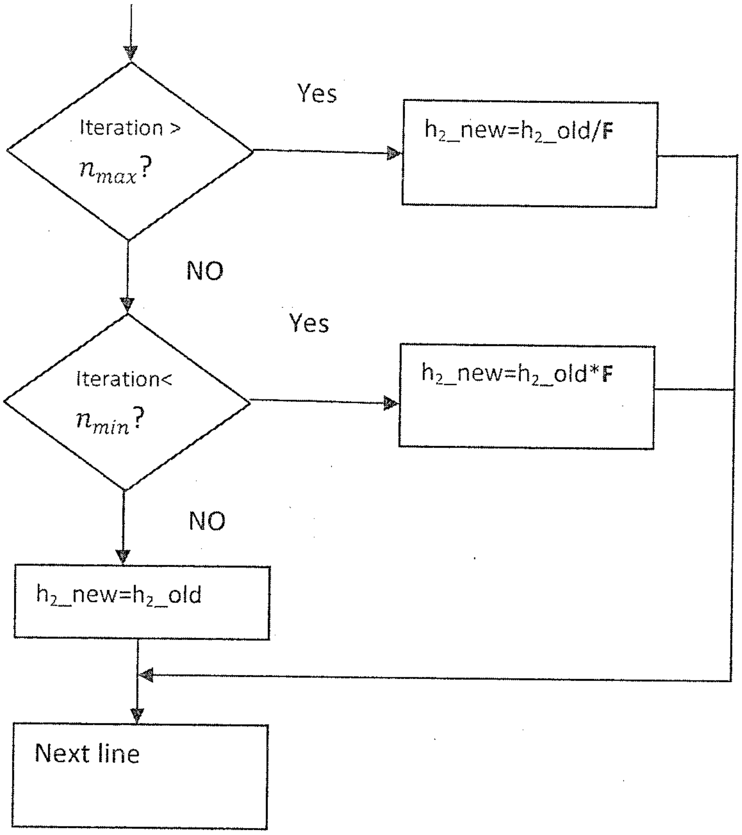


Fig.2. 26 A flow chart of adaptive time step

2.7 Vector Extrapolation

Vector extrapolation methods can be very effective for the purpose of accelerating the convergence of vector sequences. All these methods use the idea of fixed point iteration without solving the system equations in dimension *N*, the dimension of the vector space. An important property of vector extrapolation methods is the fact that they can be applied directly for solving linear and nonlinear systems. This stems from the fact that these methods do not need explicit knowledge of the sequence generator:

$$x_{n+1} = F(x_n). \tag{2.30}$$

In particular, no derivatives of F (Jacobian matrix of the function) are computed in Equation (2.30).

Suppose we need to accelerate the convergence of the general non-linear fixed point iteration in (2.30). The iteration (2.30) can be linearized in the neighbourhood of a fixed point x_s [8]. This gives;

$$x_{n+1} = F(x_n) = F\big(x_s + (x_n - x_s)\big) \approx x_s + F'(x_s)(x_n - x_s)$$

This can be written as:

$$x_{n+1} \approx F'(x_s)x_n + (I - F'(x_s))x_s$$

This shows that in the neighbourhood of a fixed point the non-linear iteration behaves approximately like the linear iteration

$$x_{n+1} = Ax_n + b \tag{2.31}$$

This suggests that if we have methods to accelerate (2.31), then these can be extended at least locally to the general non-linear iteration. In this thesis we used an extrapolation method known as Minimal Polynomial Extrapolation (MPE) [8,9].

Minimal Polynomial Extrapolation (MPE)

MPE was studied by Cabay and Jackson (1976) and considered as the most successful method of extrapolation. Sidi(1987) [7] gave an efficient implementation of the MPE. Suppose a sequence of vectors in real or complex *N*-space is generated linearly from starting point x_0.

$$x_{n+1} = Ax_n + b \ , \quad n = 0,1,2 \ldots \tag{2.32}$$

where *A* is a fixed matrix and *b* is a fixed vector, we assume that neither *A* nor *b* are known; only the sequence $\{x_n\}$ or means of generating it is known. Also we assume that 1 is not an eigenvalue of *A*, so the iteration (2.32) has a unique fixed point:

$$s = As{+}b,$$

namely,

$$\mathbf{s} = (\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{b} \tag{2.33}$$

if $\lambda < 1$ for every eigenvalue λ of A, then:

$$\mathbf{s} = \lim_{n\to\infty} \boldsymbol{x}_n$$

The objective of MPE is to determine **s** from few numbers of the terms of the sequence. If the sequence diverges, **s** is called the *anti-limit*, MPE may still determine the *anti-limit* from few terms of the sequence. The objective of MPE is to determine **s** from as few terms as possible without inverting an $N \times N$ matrix, and without any additional information about A. In a typical application, N may be quite large relative to the number of eigenvalues λ with |λ| near 1 (causing slow convergence) or greater than 1 (usually causing divergence).

MPE Algorithm [8]

Let $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$, be a given real or complex sequence of the N-dimensional column. Define:

$$u_i = \Delta x_i = x_{i+1} - x_i \quad i = 0,1,2,\ldots \tag{2.34}$$

Let $k \leq N$ be a fixed integer (to be determined), we define $N \times k$ matrices whose columns are the vectors of difference

$$U = U_K = [u_0, u_1, u_2 \ldots \ldots\ldots, u_{k-1}], \tag{2.35}$$

Note that

$$u_{i+1} = Au_i = A^{i+1}u_0 \tag{2.36}$$

We determine **s** as a weighted average of the x_i's, this weight is calculated by the coefficients of the minimal polynomial $P(\lambda)$ of A with respect to $x_i - s$, i.e. the unique monic polynomial of least degree such that

$$P(A)u_0 = 0 \tag{2.37}$$

We take $k \leq N$ to be the degree of $P(\lambda)$. Note that k can be much smaller than N. We write

$$P(\lambda) = \textstyle\sum_{i=0}^{k} c_i \lambda^i \qquad \text{and} \quad c_k = 1$$

From (2.30) and (2.31), we have:

$$\sum_{i=0}^{k} c_i\, u_i = 0\,, \tag{2.38}$$

so the vector $\mathbf{c} = (\mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_2, \dots \mathrm{c}_{\mathrm{k}-1})^{\mathrm{t}}$ of the unknown coefficients of the polynomial $P(\lambda)$ is the solution of the system equations

$$Uc = -u_k \tag{2.39}$$

The system (2.39) could have more equations than unknowns if $\mathrm{k} < N$, thus we can write the solution of the system as

$$\boldsymbol{c} = -U^{+} u_k\,, \tag{2.40}$$

where U^{+} is the Moore–Penrose pseudoinverse of U

For any k+1 consecutive terms of the sequence $\boldsymbol{x}_m, \boldsymbol{x}_{m+1}, \boldsymbol{x}_2, \dots\dots, \boldsymbol{x}_{m+k}$, we have:

$$\sum_{i=0}^{k} c_i\, u_{m+i} = \left(\sum_{i=0}^{k} c_i\right)\boldsymbol{s} \tag{2.41}$$

Algorithm 1: A summary of MPE algorithm as follows:

Starting from initial value of sequence x_0, and given the sequence generator of the form (2.32)

1. Generate the sequence $\boldsymbol{x}_1, \boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{k+1}$
2. Calculate U, u_k using Equations (2.34) and (2.35)
3. Calculate **c** using Equations (2.39) and (2.40)
4. Calculate **s** using Equation (2.41)

In practice, when dealing with nonlinear sequences, the algorithm above is extended by *cycling* to generate a sequence s_0, s_1, s_2 ... of approximation to **s** [8].

Algorithm 2: Nonlinear Extrapolation Algorithm with Cycling.

```
1. Set $s_0$ = $x_0$, and $i$ = 1
2. Generate $k$+1 vectors $\mathbf{x}$ for the MPE
3. Apply MPE to compute $s^*$
4. Set $s_i$ = $s^*$, increase $i$ by 1, set $x_0$= $s^*$, and return to step 2.
```

Each time this loop is executed is called a cycle. It's difficult to find the right degree *k* of the polynomial. However, the cycling technique is the most practical technique to deal with nonlinear systems [8].

Chapter 3

Method to obtain a good initial guess in HB analysis in oscillators.

3.1 Introduction

As mentioned in Chapter 1, regular HB is very sensitive to the initial guess of the solution. Regular HB analysis for oscillators may converge to the unstable equilibrium DC point or may not converge at all if a good initial guess of the solution is not provided. It can be very difficult to obtain such good initial guess of the oscillator solution, in particular, the guess of the oscillation frequency. In this chapter we will present an approach to find a good initial guess for HB by applying Minimal Polynomial Extrapolation (MPE) to the solution of Warped Multi-time Partial Differential Equation (WaMPDE). In Ref. [6], WaMPDE has been proven to be a fast and a robust method to find the transient and steady state of oscillators. The MPE has the ability to accelerate finding the steady-state of the solution obtained by WaMPDE. If a nonlinear system (i.e. oscillators) is bound to reach steady-state, MPE will accelerate this process (finding *the* solution *limit).* In our approach, the extrapolation is applied along the real time (τ_2). The MPE is expected to use a few solutions of the WaMPDE along the warped time τ_2 and then predict the limit of these solutions. Therefore, the number of solutions along τ_2 should decrease before this solution is applied as initial guess to the regular HB. Our choice of using MPE arises from the fact that MPE doesn't require calculating the derivatives (Jacobian) of the nonlinear equation to find the solution, while WaMPDE needs to calculate and factor the Jacobian matrix for each time step along the real time τ_2. Calculating and factoring the Jacobian can be computationally expensive; therefore extrapolation may result in computational savings. In the next section, we will present the algorithm of our approach.

3.2 The proposed algorithm

We first illustrate how the MPE algorithm is applied to WaMPDE. The general nonlinear system formulated with Fourier transform is given in Equation (2.27):

$$(G + Cji\omega_0)X^i + C\frac{dX^i}{d\tau_2} + ji\omega_0 Q^i + \frac{d\,Q^i}{d\,\tau_2} + I^i - S^i = 0$$

The WaMPDE solves the nonlinear equation above along the warped time for each time step along the real time τ_2. The solution sequences are along the warped time. These sequences are composed of the nodal voltages vector and the fundamental frequency (ω_0). Each voltage contains a harmonics and 1 dc component. Note for adaptive HB a varies for all nodal voltages. Now we show how to construct the vector sequences which will be extrapolated by the MPE method.

i. Starting τ_2=0, run the WaMPDE for *k* steps along the real time τ_2.

ii. construct a vector sequence R:

$$R = \begin{vmatrix} X(0) & X(h_2) & X(2h_2)\ldots\ldots & X((k-1)h_2) \\ \omega_0(0) & \omega_0(h_2) & \omega_0(2h_2) & \omega_0((k-1)h_2) \end{vmatrix}$$

Where h_2 is the time step along τ_2.

iii. Then Algorithm 1 of the MPE method mentioned in Section 2.7 is used to extrapolate the vectors k of the sequences (*R*) or in other words, accelerating capturing the steady-state.

Next we will show the algorithm of the proposed method:

The following provisions used in Ref. [6] (mentioned in Section 2.6) are used to improve the method efficiency:

- Adaptive HB
- Adaptive Newton tolerance

Unlike the method in Ref. [6], the method proposed in this work uses constant time steps; however adaptive time step is also used for comparison. To best utilize the interactions between MPE and WaMPDE; the extrapolation is done when the transients have started to decay. This is accomplished by setting some appropriate tolerance ϵ (e.g. 20%), and running WaMPDE until the error as calculated in Equation (2.28) and Equation (2.29) is smaller than the tolerance ϵ. When the error is smaller than ϵ, at this point along τ_2 the transient may be started to decay. At this point MPE is used to extrapolate the *k* samples along τ_2 and the result is used as a new initial guess for the WaMPDE. The ϵ is further decreased and the process is repeated in this fashion until the desired tolerance ϵ_{ds} (e.g.0.1 %) is achieved. This is the point where the WaMPDE is stopped and the regular HB is engaged. The flow chart in Fig. 3.1 explains the algorithm. Here the simulation process is sectioned and every section is called *a run* (1st run, 2nd run.........Mth run); runs represent the number of times the WaMPDE is executed. Note the number or runs = number of times the WaMPDE is extrapolated - 1. Here (ϵ_1, ϵ_2 , ... ,ϵ_M) represent the tolerances associated with each corresponding run. Note ϵ_M = ϵ_{ds}. Every successive tolerance is decreased by dividing it by some factor . The time step is increased by a factor **p** for the next run. For the last run, The WaMPDE is executed with initial guess from the last time the extrapolation is used until the error is less than the desired tolerance $\boldsymbol{\epsilon_{ds}}$. At that point the transient evolution is stopped and the regular HB is engaged with the final solution of the WaMPDE used as the HB initial guess. Regular HB takes a few additional Newton iterations to determine the final solution. Using adaptive HB allows nodal voltage to have different number of harmonics. Thus at the end of each run, zeros are added as necessary after the last harmonics of each voltage to even the length of R.

Simulation parameters for a general oscillator:

- Number of runs: no specific rule of how many runs shall be in the algorithm. However, two runs can achieve satisfactory results for smaller circuits..
- The tolerances(ϵ_1, ϵ_2, ..., ϵ_M) must be wisely selected: For example if two runs are used; the first tolerance (ϵ_1) would be in the range about 10% - 30%, second tolerance ϵ_2 = ϵ_{ds} in the range about 0.1% - 5.0%. The difference between ϵ_1 and ϵ_{ds} must not be large so that WaMPDE in the last time wouldn't run for long time before HB is engaged. Special attention must be directed to the choice of the runs tolerances. The algorithm must check how many solutions it produces for each run. The first tolerance is the most critical choice, if it is selected small, this will ensure the MPE will produce a solution closer to the steady-state. However, choosing a very small tolerance (i.e. 0.1%) for the first run might decrease the benefit of using MPE as opposed to using the WaMPDE alone. Choosing a large tolerance (i.e. 50%) will also decrease the MPE efficiency, since the extrapolation contains rapid transients.
- The most sensitive simulation parameter is the time step increase factor **p** for each run. The difficult part is determining the optimum **p, p** accelerates the solution in expense of reliability. The simulation starts with small time step, after the first run, the solution is likely starting to decay, therefore a larger time step might be to be used for the next run. The time step increase factor depends on the initial time step and the tolerance of WaMPDE for each run. Smaller tolerance and smaller initial time step could allow a bigger increase in **p** in the following run.
- It is better to extrapolate only the last few samples (e.g. extrapolate the last four samples or the last third of the total samples). In particular the first run can contain high nonlinear values, the simulation shall perform better if only a few of the final values are extrapolated.

The MPE method accelerates the work of the WaMPDE in Ref. [6]. In the next chapter, we will show some simulation to evaluate the performance of the method.

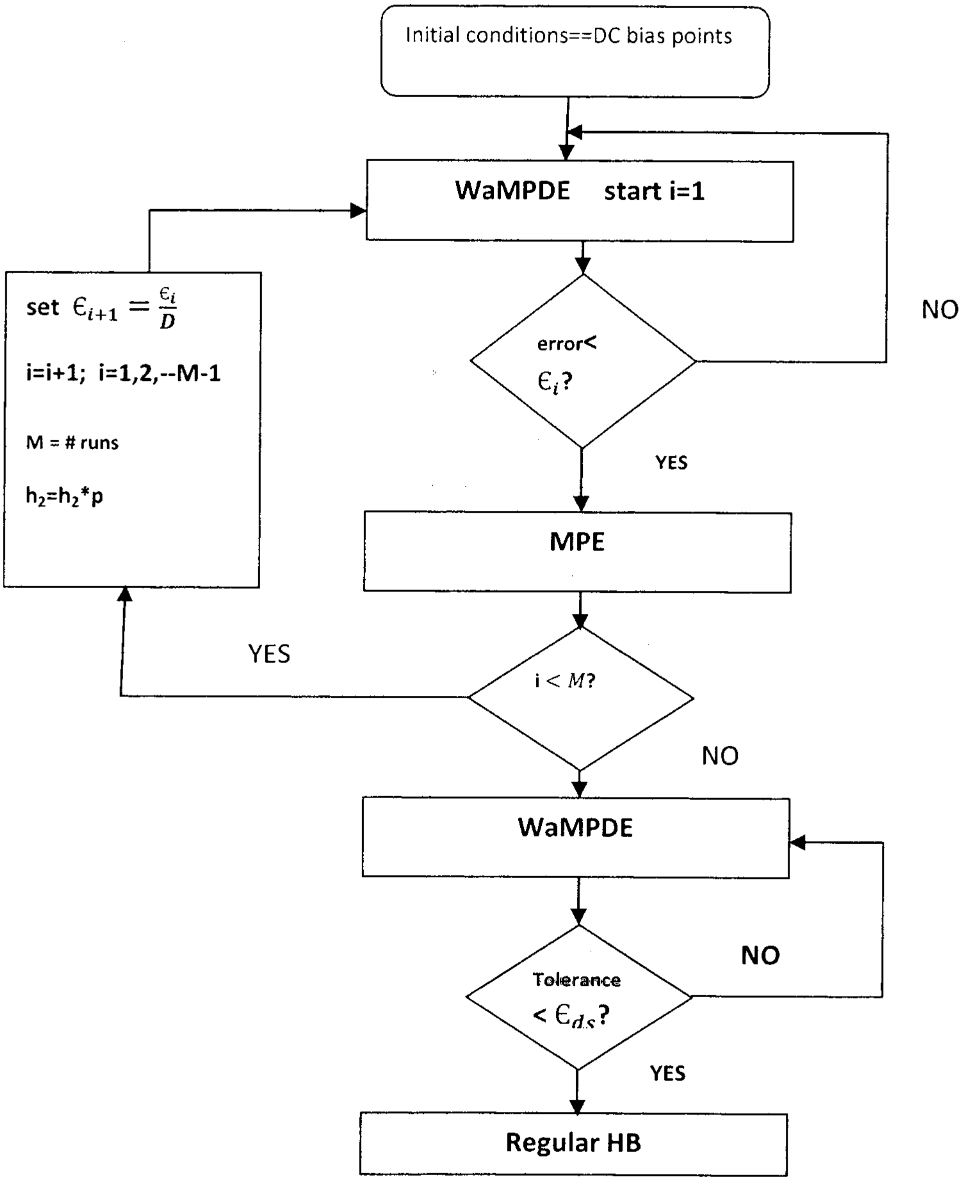


Fig.3. 1 The flow chart the proposed method

Chapter 4

Simulations

4.1 Introduction

The authors of Ref. [6] have improved capturing the steady-state response of oscillators using WaMPDE in conjunction with adaptive harmonic balance. The method proposed in Chapter 3 further improves the method in Ref. [6]. In this chapter, we present simulation results for two oscillators: an LC-tuned bipolar oscillator and a Colpitts oscillator. Results of the proposed method are presented and compared by varying simulation parameters and also compared with results in Ref. [6]. All the simulations are performed with MATLAB on 2.4 GHz Pentium dual-core computer.

4.2 LC-tuned bipolar oscillator

Fig. 4.2.1 shows the schematic of the LC-tuned bipolar oscillator [13]. The parameters of this circuit are: C_1 = C_2 =33 pF, C_3 =3.17 pF, C_C =560 pF, L_1 =nH, R_F = 680 Ω, R_B =100 KΩ, R_C =1.2 kΩ and V_{dd} =10 V. In order to formulate the circuit nodal equations, we replace the transistor with the large signal model shown in Fig. 4.2.2, the inductor with the equivalent circuit shown in Fig. 4.2.3, and the Thevenin source with Norton source in Fig. 4.2.4. We substitute those models in Fig. 4.2.1.

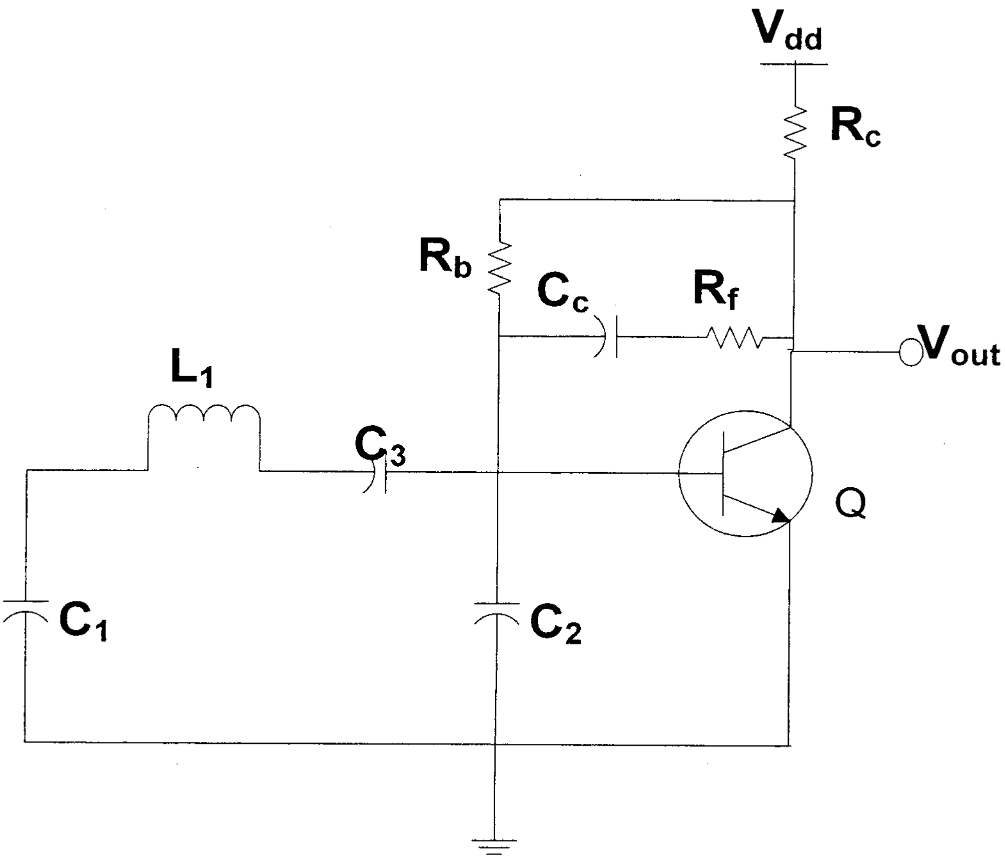


Fig.4.2. 1 LC-tuned bipolar oscillator schematic diagram

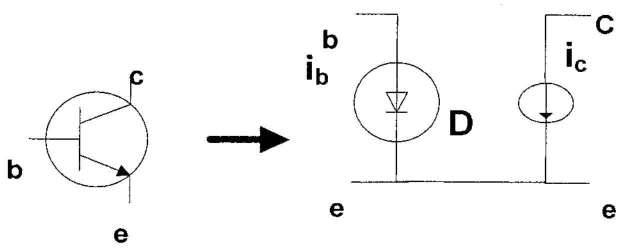


Fig.4.2. 2 Large signal model of the transistor

The base current is $I_b = \frac{1}{R_f}\left(e^{\frac{V_{BE}}{V_T}} - 1\right)$, the collector current is $I_c = BF \times I_b$, with saturation current $I_s = 0.01pA$, thermal voltage $V_T = 26\ mV$ and forward beta $BF = 100$.

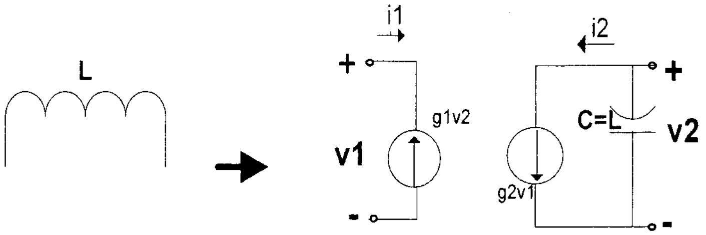


Fig.4.2. 3 Inductor equivalent circuit

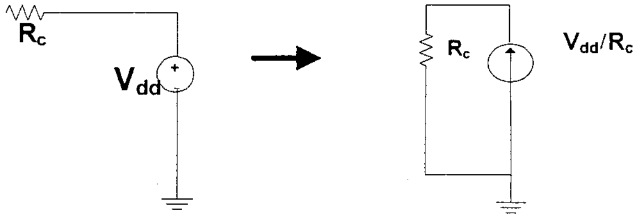


Fig.4.2. 4 Norton equivalent of the DC source

The resulting circuit is shown in Fig. 4.2.5 where all nodes are defined ($x = |1^{st}2^{nd}\ 3^{rd}\ 4^{th}\ 5^{th}|^T$). Then we formulate the nodal equations of the circuit according to Equation (2.26):

$$Gx(t) + C\frac{dx(t)}{dt} + \frac{dQ(x(t))}{dt} + I(x(t)) = S(t)$$

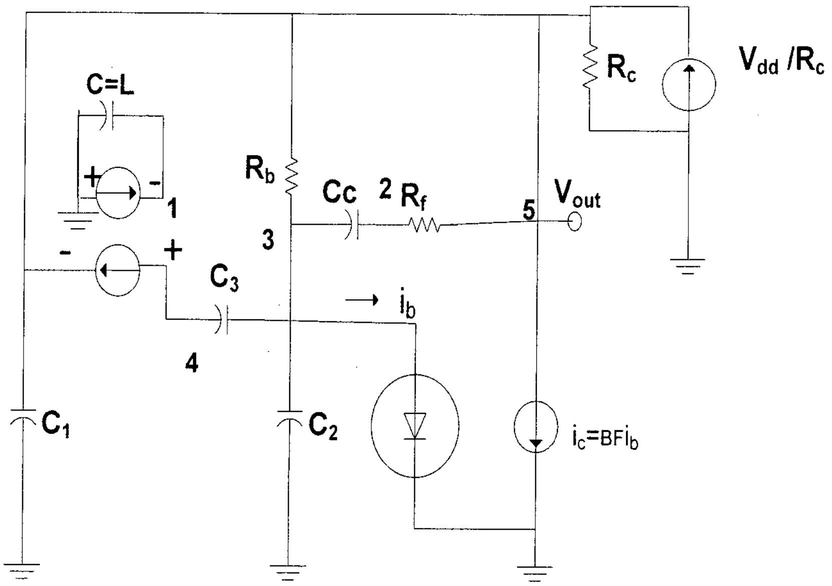


Fig.4.2. 5 LC-tuned bipolar oscillator equivalent model

Each matrix is defined by:

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 & -1 \\ 0 & G_f & 0 & 0 & -G_f \\ 0 & 0 & G_b & 0 & -G_b \\ -1 & 0 & 0 & 0 & 0 \\ 1 & -G_f & -G_b & 0 & G_f + G_c + G_b \end{bmatrix}, C= \begin{bmatrix} L & 0 & 0 & 0 & 0 \\ 0 & C_c & -C_c & 0 & 0 \\ 0 & -C_c & C_c + C_2 + C_3 & -C_3 & 0 \\ 0 & 0 & -C_3 & C_3 & 0 \\ 0 & 0 & 0 & 0 & G_1 \end{bmatrix},$$

$$I(x) = \begin{bmatrix} 0 \\ 0 \\ I_b \\ 0 \\ I_c \end{bmatrix}, \quad S = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \frac{V_{dd}}{R_c} \end{bmatrix}$$

with $G_f = \frac{1}{R_f}, \quad G_b = \frac{1}{R_b}, G_c = \frac{1}{R_c}$

The DC bias points are calculated to be:

$$\begin{cases} X_1 = \ \ 0\,V \\ X_2 = 4.9\,V \\ X_3 = 0.7\,V \\ X_4 = 4.9\,V \\ X_5 = 4.9\,V \end{cases}$$

4.2.1 Simulation results and discussion:

The simulation starts with the DC bias points. The excitation current is injected into the base node with a magnitude of 10µ A. The simulation is set to two runs; first run tolerance ϵ_1 is set to 0.6 and second run tolerance ϵ_2 is set to $\epsilon_1/100$ =0.0006. Using constant time steps of 0.2 ns, the simulation produced 14 samples of solutions along τ_2 with 56 Newton iterations. Extrapolating the last 4 samples, increasing the constant time step for the second run to 0.02 s and using the extrapolated value as the new initial value, we obtained 4 samples of solution along τ_2 with 9 Newton iterations to obtain ϵ_2. The final solution from the second run is fed to regular HB as initial value; the regular HB needed only 3 Newton iterations to capture the steady-state . The fundamental frequency is found to be 308.709 MHz. The total simulation time was 5.6 seconds, 0.001 seconds of which is the simulation time of the MPE.

Fig. 4.2.6 and Fig. 4.2.7 show the simulation results of the output voltage (voltage at node 5) and fundamental frequency during the first run. It is obvious that the fundamental frequency tend to reach a constant value quickly while the voltages are still in transient state (only the output voltage is shown). Fig. 4.2.8 and Fig. 4.2.9 show the simulation results of the second run; the voltages tend to be steady, and the fundamental frequency changes very little and then reaches a constant value. Simulation of the output voltage of the regular HB is shown in Fig. 4.2.10. Fig. 4.2.11 shows the steady-state solution of HB together along with the final multi-time solution, which indicates that the two solutions are very similar. These results are summarized in Table. 4.2.1.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
1s run	14	56	5.6 seconds
2nd run	4	9	
Regular HB	NA	3	

Table 4.2. 1 A summary of the simulation results

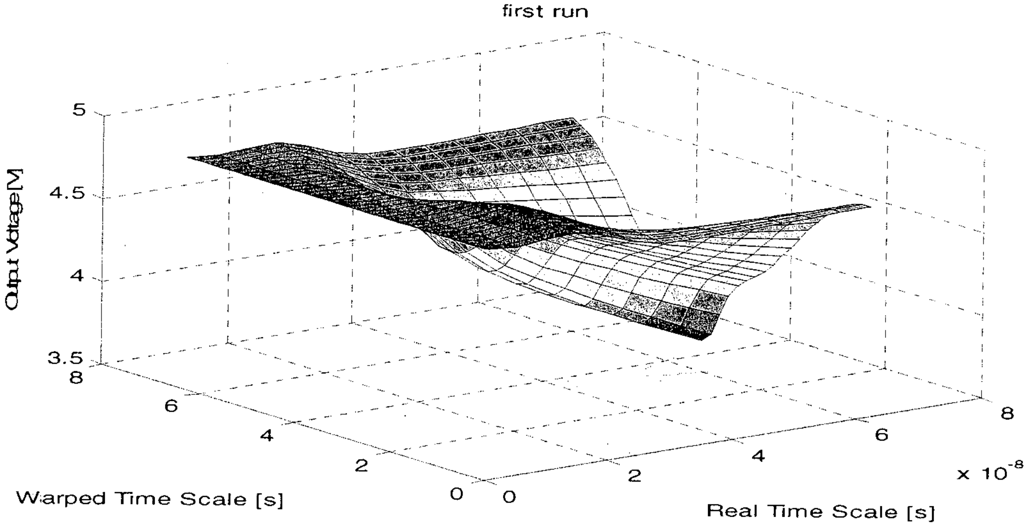


Fig.4.2. 6 The output voltage during the first run

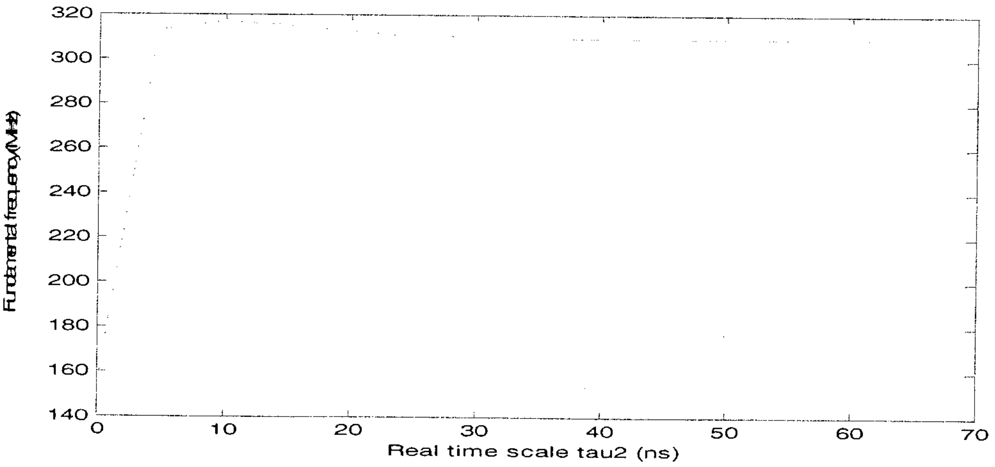


Fig.4.2. 7 The fundamental frequency during first run

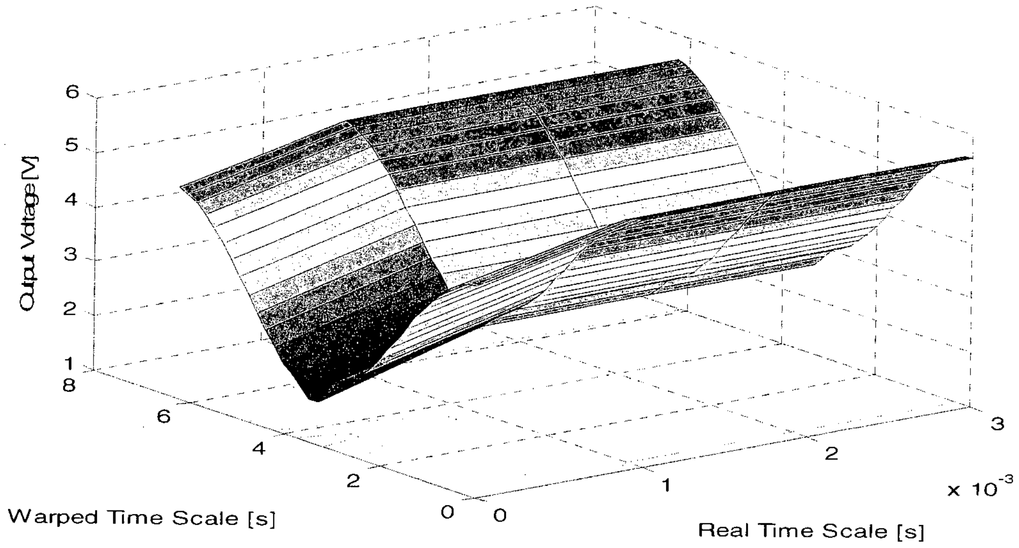


Fig.4.2. 8 The output voltage during second run

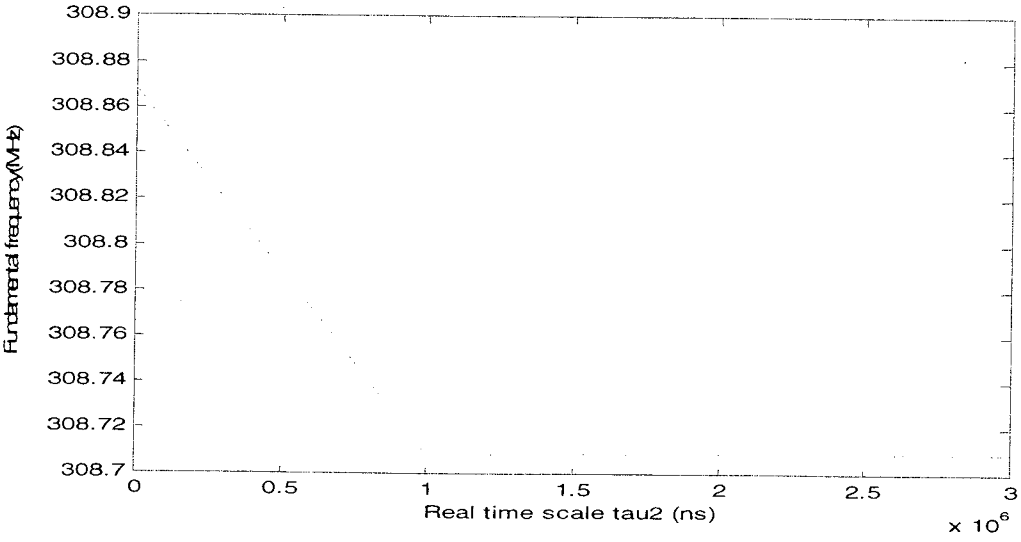


Fig.4.2. 9 The fundamental frequency during second run

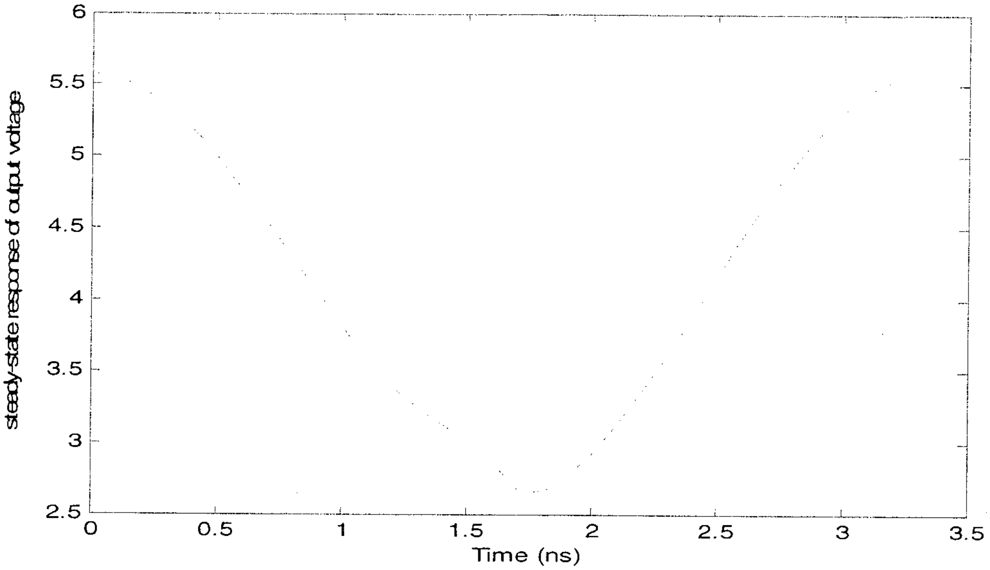


Fig.4.2. 10 The regular HB steady state voltage

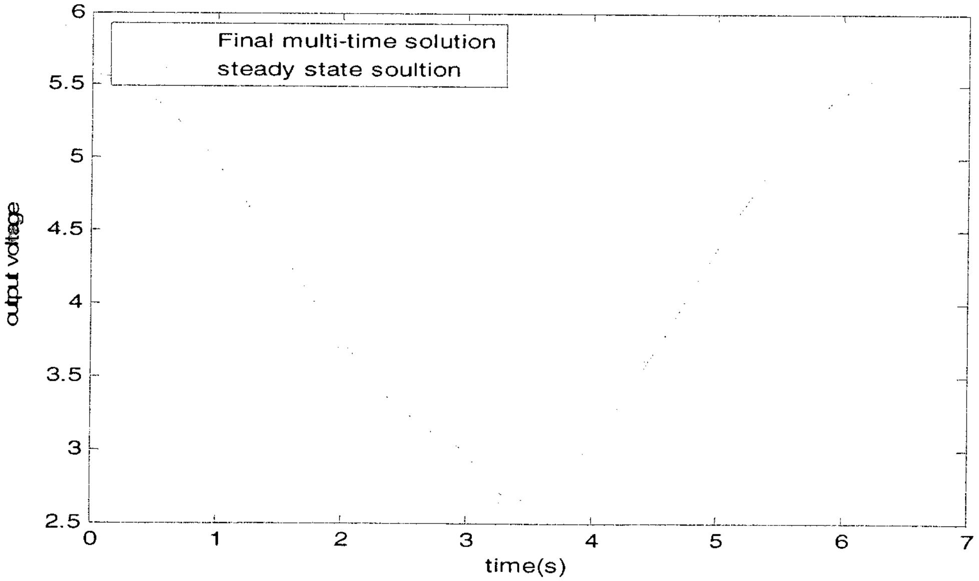


Fig.4.2. 11 Steady-state solution compared to final multi-time solution

Comparison of Various Simulations

1. Repeating the same simulation but varying the number of samples for extrapolation:

- Extrapolating the last 8 samples and the last 5 samples produced the same results as extrapolating 3 lines.
- Extrapolating all the 14 samples produced the same results but the number of Newton iterations in the second run was 13 (an increase by 3).

2. (a) Repeating the same simulations but set ϵ_1 =0.2, ϵ_2= 0.0002. Results in Table. 4.2.2

   (b) Repeating the same simulations but set ϵ_1 =0.2, ϵ_2= 0.0001. Results similar to part (a)

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total simulation time
1s run	23	69	6.4 seconds
2nd run	4	7	
Regular HB	NA	2	

Table 4.2. 2 Results with varying tolerances

   (c) Repeating the same simulation but set ϵ_1 =0.1, ϵ_2 =0.00001. Results in Table. 4.2.3

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total simulation time
1s run	27	76	6.6 seconds
2nd run	4	7	
Regular HB	NA	2	

Table 4.2. 3 Results with varying tolerances

3. Repeating the simulation with time step = 2 ns for both runs. Results in Table. 4.2.4

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total simulation time
1s run	28	81	7.1
2nd run	14	17	
Regular HB	NA	5	

Table 4.2. 4 Results with varying time step

4. Repeating the same simulation but with adaptive time step with a factor= 1.8. The total number of samples is 30 and the total number of Newton iterations is 93. Results in Table. 4.2.5

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
1s run	19	68	5.6 seconds
2^{nd} run	11	22	
Regular HB	NA	3	

Table 4.2. 5 The proposed method with adaptive time step

5. The simulation in Ref. [6] which uses adaptive time step with factor=1.8, initial time step=2 ns and Newton threshold values are 5 and 10 is shown in Table. 4.2.6. The simulation produced 34 samples and a total of 123 Newton iterations. The output voltage of the multi-time solution and the steady-state solutions are shown in Fig. 4.2.12 and Fig. 4.2.13 respectively.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
First step	34	121	5.5 seconds
Regular HB	NA	3	

Table4.2. 6 Simulation results of method in Ref [6]

6. For some choices of the initial time step along τ_2; the method in Ref. [6] and the proposed method produce an unstable DC operating point solution. However, the proposed method finds the steady-state solution for some of these choices. For example the method in Ref. [6] doesn't find the steady-state solution for initial time steps: 0.01 ns, 0.0125 ns, 0.017 ns and 0.02 ns while the proposed method finds the steady-state solution for 0.02ns and 0.017 ns and fails to find it for 0.01ns and 0.0125 ns

In this circuit, it is clear the proposed method has improved the method in Ref. [6] in terms of reducing the number of Newton iterations needed to capture the steady-state, thus resulting in saving the CPU memory. Using constant time step; the reducing in Newton iterations varies from 40% to 50%. In this circuit, the proposed method performs much better using constant time step as opposed to using adaptive time step. The saving when adaptive time step is used with the proposed method is about 25%. In terms of simulation time, there is no considerable difference between the method in Ref. [6] and the proposed method. The proposed method also reduces the number of solution samples. The simulation shows it is better to extrapolate the last few samples, extrapolating all the samples might slow the process; since the first samples are highly nonlinear (transient). The simulation time attributed to the MPE is very small.

Method [6] produced the best results with a time step factor F=1.8, it needs 1394 Newton iterations and 40 seconds with no adaptive time step (i.e. time step factor F=1). Choosing the adaptive time step factors is sensitive, for example if the time step factor is 2; the method produces the unstable DC operating point solution. Thus the simulation might need to run for few times with modifying the adaptive time step factors to get the right solution. Also if the number of Newton iterations of the solution does not lie on the constraints (i.e. n_{min} and n_{max}); the adaptive time step algorithm won't be used. The proposed method is as well sensitive to simulation parameters (i.e. number of extrapolated samples, tolerances, time step for each run). It is difficult to get the optimum results in the first try. However, the results in this oscillator might indicate the proposed method might be slightly more reliable than the method in Ref. [6]; it might not produce the optimum results but the right solution seems obtainable for wider range of simulation parameters.

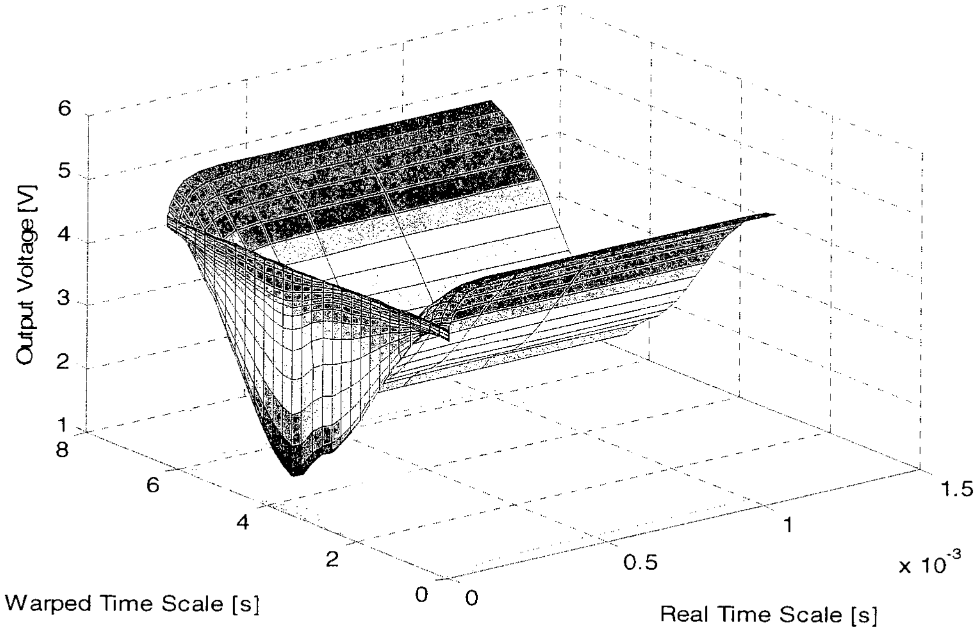


Fig.4.2. 12 The multi-time output voltage in Ref [6]

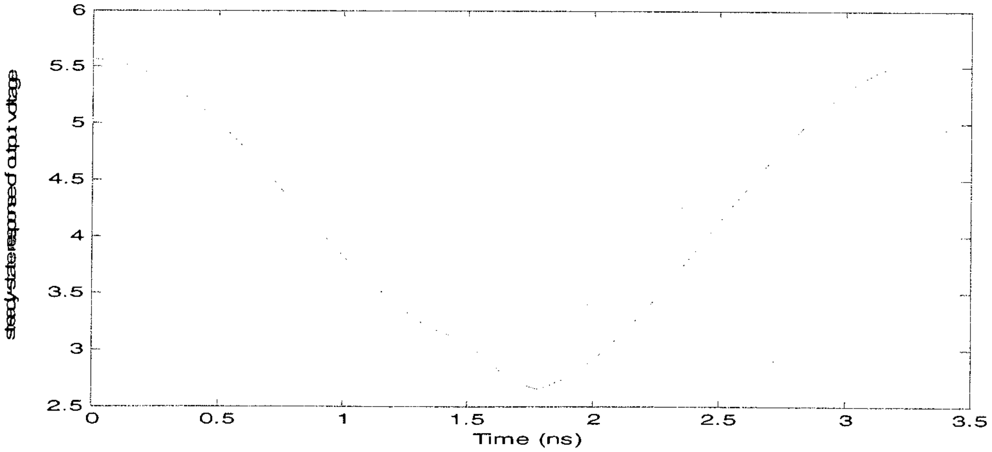


Fig.4.2. 13 The regular HB steady-state voltage in [6]

4.3 Colpitts Oscillator

Fig. 4.3.1 shows the schematic of the Colpitts oscillator [14]. The LC tank in this oscillator uses a capacitive voltage divider. The values of the circuit parameters are: C_1 = C_2 =2 pF, C_e =100 pF, C_C =400 pF, L_1 = 1μH, R_1 = 8 kΩ, R_2 =2 KΩ, R_C = 2.4 kΩ, R_e = 1.3 kΩ, V_{cc} =11 V, BF =100, and BR =1.

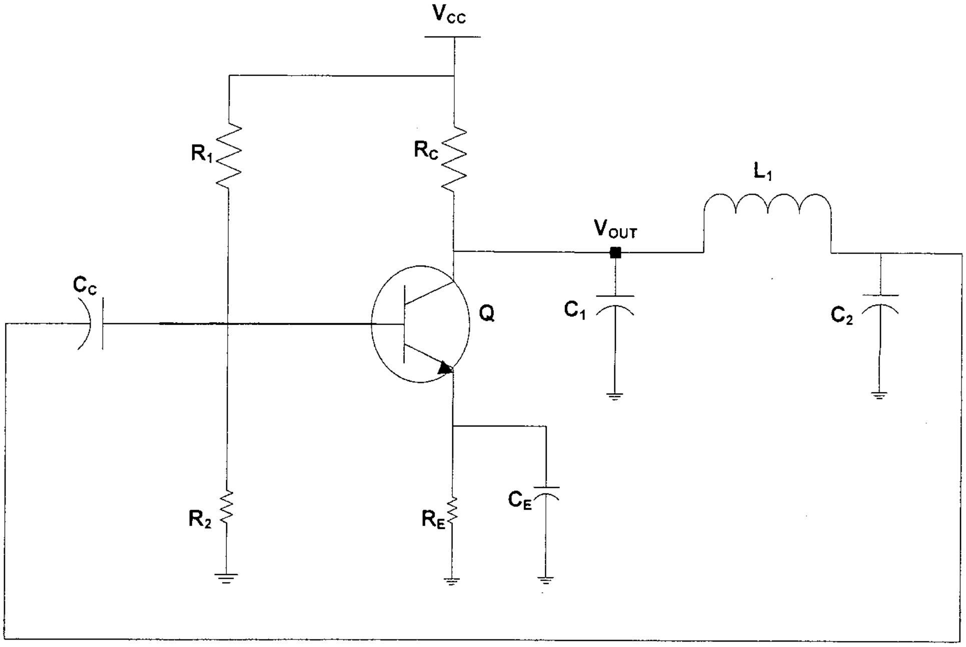


Fig.4.3. 1 Colpitts oscillator schematic diagram

To consider the transistor saturation, we replace the transistor with the Ebers-Moll transistor model shown in Fig. 4.3.2. The Ebers-Moll model has two diodes and two current sources which model all operation regions of the transistor: forward active mode, reverse active mode, saturation and cut-off mode. Ebers-Moll allows relating the emitter, base and collector current to the forward and reverse currents and transport factors. The diode currents and source currents are given by:

$$I_{ubc} = \frac{I_S}{BR}\left(e^{\frac{V_{BC}}{V_T}} - 1\right)$$

$$I_{ube} = \frac{I_S}{BF}\left(e^{\frac{V_{BE}}{V_T}} - 1\right)$$

$$I_{uec} = BR \times I_{ubc}$$

$$I_{uce} = BF \times I_{ube}$$

The saturation current I_S =0.01 pA. Thermal voltage V_T =26 mA. Replacing all equivalent models in Fig. 4.3.1 we have resulting equivalent circuit shown in Fig. 4.3.3 with all the nodes defined.

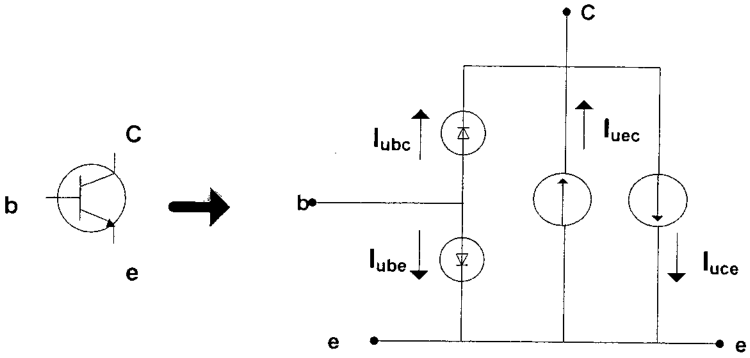


Fig.4.3. 2 The Ebers-Moll model of the bipolar transistor

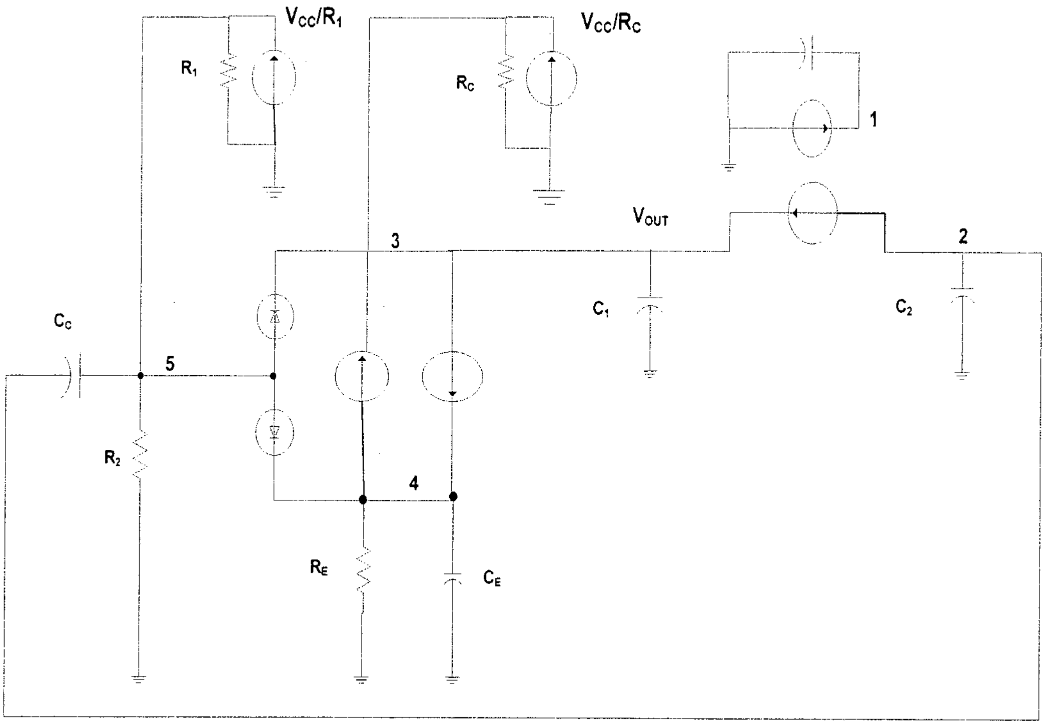


Fig.4.3. 3 Colpitts oscillator equivalent circuit

Then we formulate the nodal equations of the circuit according to Equation (2.26):

Each matrix is defined by:

$$
G = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & G_C & 0 & 0 \\ 0 & 0 & 0 & G_e & 0 \\ 0 & 0 & 0 & 0 & G_1 + G_2 \end{bmatrix}, \quad C = \begin{bmatrix} L & 0 & 0 & 0 & 0 \\ 0 & C_2 + C_c & 0 & 0 & -C_c \\ 0 & 0 & C_1 & 0 & 0 \\ 0 & 0 & 0 & C_e & 0 \\ 0 & -C_c & 0 & 0 & C_c \end{bmatrix},
$$

$$
I(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ I_{uce} - I_{uec} - I_{ubc} \\ -I_{ube} - I_{uce} + I_{uec} \\ I_{ubc} + I_{ube} \end{bmatrix}, \quad S = \begin{bmatrix} 0 \\ 0 \\ -\frac{V_{cc}}{R_c} \\ 0 \\ -\frac{V_{cc}}{R_1} \end{bmatrix}
$$

with $G_1 = \frac{1}{R_1}$, $G_2 = \frac{1}{R_2}$, $G_e = \frac{1}{R_e}$, $G_c = \frac{1}{R_c}$

The DC bias points are calculated to be:

$$\begin{cases} X_1 = \ \ 0\,V \\ X_2 = 8.4\,V \\ X_3 = 8.4\,V \\ X_4 = 1.4\,V \\ X_5 = 2.2\,V \end{cases}$$

4.3.1 Simulation results and discussion:

The simulation starts with DC bias points. The excitation current is injected into the base node with a magnitude of 10µ A. The simulation is set to two runs; first run tolerance ϵ_1 is set to 0.8, and the second run tolerance ϵ_2 is set to $\epsilon_1/100$ = 0.008. Using constant time steps of 10 ns during the first run, the simulation produced 6 samples of solutions along τ_2 with 30 Newton iterations. Extrapolating the last 4 samples, increasing the constant time step to 40 ns for the second run and sending the extrapolated value as the new initial value, we obtained 83 samples of solutions along τ_2 with 269 Newton iterations to obtain ϵ_2. The final solution from the second run is fed to regular HB as initial value; the regular HB needed only 7 Newton iterations to capture the steady-state. The fundamental frequency is found to be 5.036 MH. The total simulation time was 16.2 seconds, 0.001 of which is the simulation time of the MPE.

Fig. 4.3.4 and Fig. 4.3.5 show the simulation results of the output voltage (voltage at node 3) and fundamental frequency during the first run. It is obvious that the fundamental frequency changes rapidly during the first run and the output voltage is still not oscillatory. Simulations of the output voltage and the fundamental frequency during the second run are shown in Fig. 4.3.6 and Fig. 4.3.7, respectively. The output voltage still experiences no oscillation in the beginning of the second run and then starts to oscillate. Also, the fundamental frequency has not reached steady value in the beginning of this run but it reaches a point close to steady value in this run. Simulation of the output voltage of the regular HB is shown in Fig. 4.3.8. Fig. 4.2.9 compares the steady-state solution of HB with the final multi-time solution, which shows they are close but not as similar as noticed in the LC-tuned bipolar oscillator (Fig. 4.2.11). A closer solution can be obtained if the second tolerance is decreased but this will result in increasing the computation effort. Decreasing the second tolerance is not important since the regular HB needed only 7 Newton iterations to capture the steady-state solution. These results are summarized in Table. 4.3.1.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time seconds
1s run	6	30	16.2
2nd run	83	269	
Regular HB	NA	7	

Table. 4.3. 1 Simulation summary

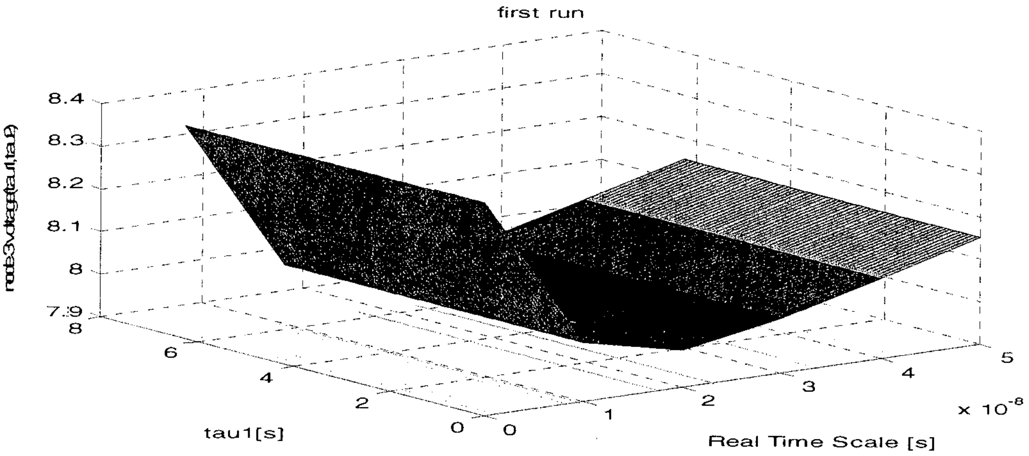


Fig.4.3. 4 The output voltage during first run

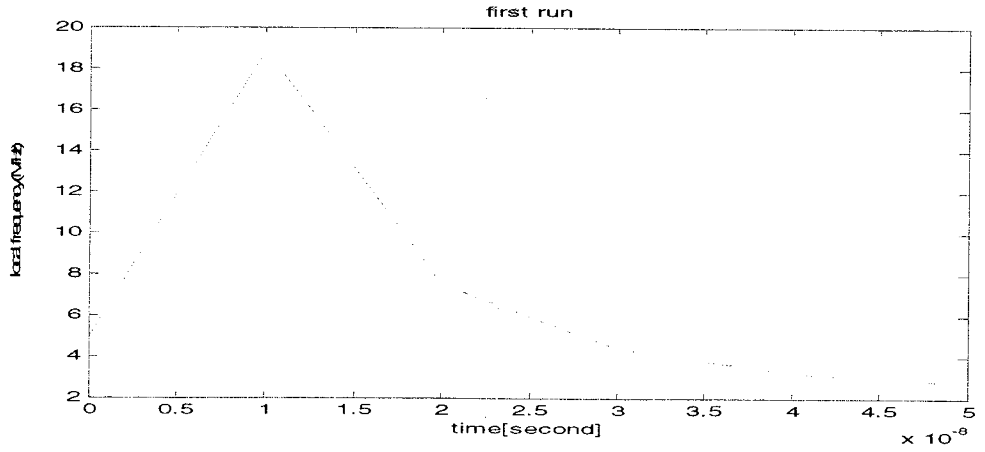


Fig.4.3. 5 The fundamental frequency during first run

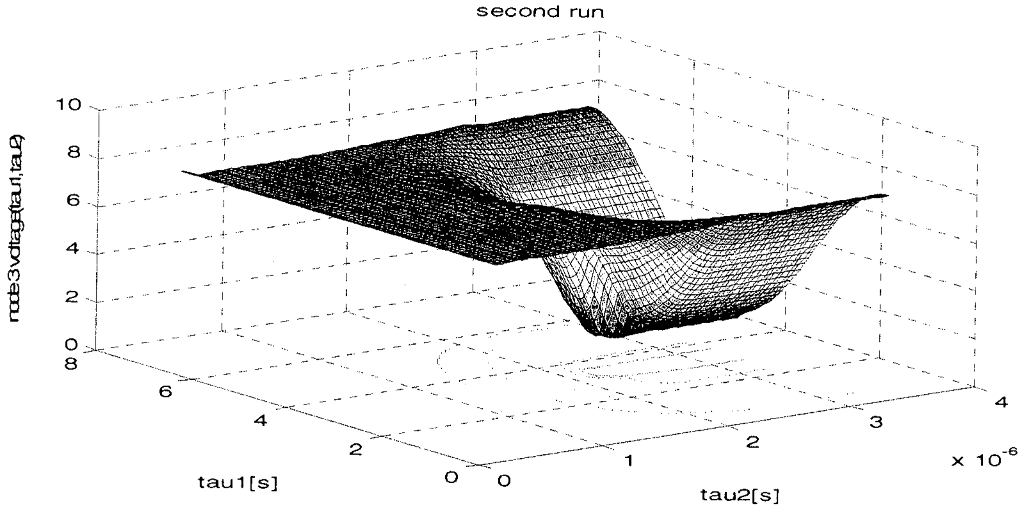


Fig.4.3. 6 The output voltage during second run

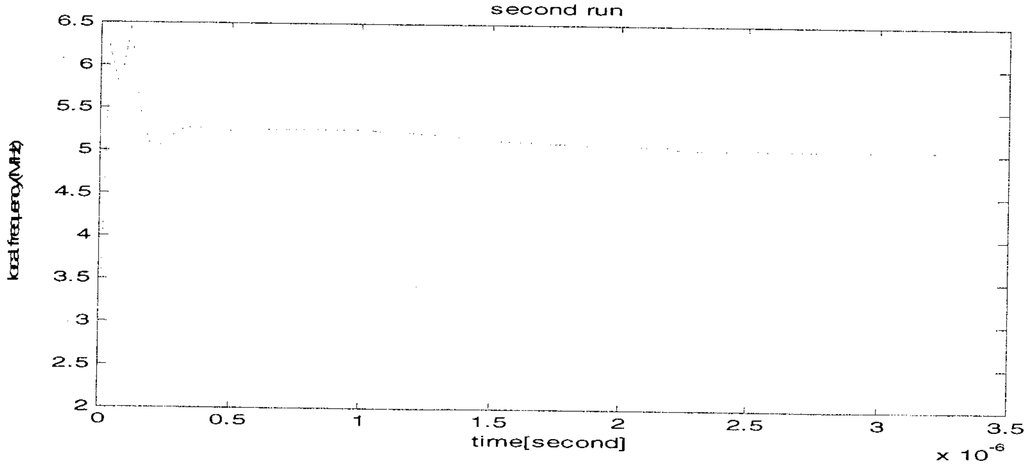


Fig.4.3. 7 Fundamental frequency during second run

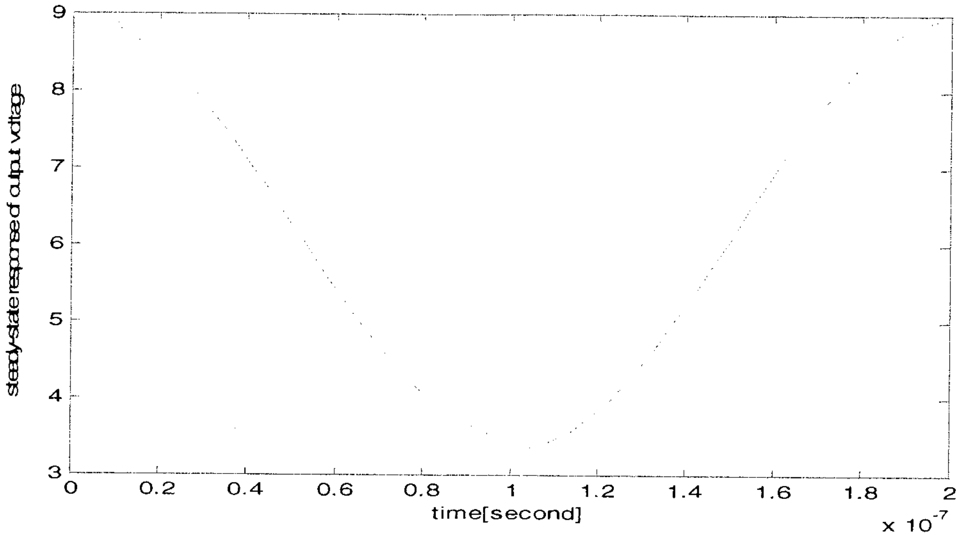


Fig.4.3. 8 Steady-state solution of regular HB

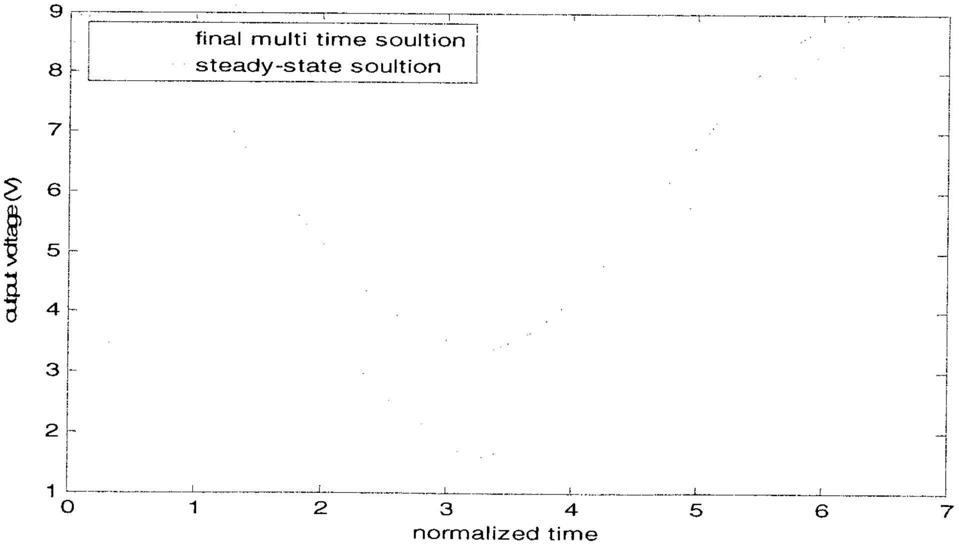


Fig.4.3. 9 Steady-state solution compared to final multi-time solution

Comparison of Various simulations

1. Repeating the same simulation but varying the number of samples to be extrapolated:

- Extrapolating the last 3 samples produced the same results
- Extrapolating all the 6 samples of the first run produced the same results but the number of Newton iterations in the second run increased to 800. This large increase might be due to the fact that the first samples have no oscillations and are all extrapolated. Only the output voltage is shown here, but most of the nodal voltages don't start oscillating for many time steps along the real time; Colpitts is known for its long transient [6].

2. Repeating the same simulation with the same time steps for both runs, both equal to 10 ns. Results in Table. 4.3.2.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total simulation time
1s run	6	30	26 seconds
2nd run	119	421	
Regular HB	NA	7	

Table. 4.3. 2 Simulation with the same time step for both runs

3. (a) Repeating the same simulation but set ϵ_2=0.1 and ϵ_2 is kept the same (i.e. 0.8). Results in Table. 4.3.3.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total simulations time (seconds)
1s run	6	30	15.4
2nd run	78	259	
Regular HB	NA	7	

Table. 4.3. 3 Simulation with varying tolerances

(b) Setting ϵ_1=0.6 and ϵ_2= 0.01. The Results are similar to part (a)

4. Repeating the same simulation but with adaptive time step with a time step factor= 1.5 for both runs. The total number of samples is 66 and the total number of Newton iterations is 371. Results in Table. 4.3.4

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
1s run	6	30	16.6 seconds
2^{nd} run	60	331	
Regular HB	NA	10	

Table. 4.3. 4 The proposed method with adaptive time step

Simulation with three runs; ϵ_1=0.8, ϵ_2=0.4 and ϵ_3= 0.0004. Using constant time steps of 10 ns during the first run, 40 ns for the second run and 160 ns for the third run. The last 4 samples are extrapolated in each run. The results are shown on Table. 4.3.5. The output voltage and fundamental frequency are shown in Fig. 4.3.10, and Fig. 4.3.11 for the first run, Fig. 4.3.12, and Fig. 4.3.13 for the second run and Fig. 4.3.14, and Fig. 4.3.15 for the third run. Fig. 4.3.16 shows the steady-state and final multi- time solutions are close. In the first and second run, the voltage still doesn't experience oscillations, however, the output voltage experience unwanted oscillations equal to the fundamental frequency along the real time. In start of the third run, the output voltage and fundamental frequency became steady. However, the other voltages have not reached the steady-state (not shown here). Comparing the results with the results of two runs in Table. 4.3.1; the number of samples and Newton iterations increased when using three runs. Using three runs for this circuit doesn't appear to be helpful to add any benefits than using two runs. The nature of this oscillator is that it doesn't oscillate for some time and when it does, it reaches steady-state very fast. This nature reduces the benefits of the MPE methods. For example if ϵ_2 is reduced to 0.1, the WaMPDE will run and stop at a point τ_2 close to the steady-state, here at this point the error reduces by a factor of 100 in two successive samples along τ_2. Results are summarized in

Table. 4.3.5

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
1s run	6	30	21.3 seconds
2^{nd} run	66	221	
3^{rd} run	41	113	
Regular HB	NA	6	

Table. 4.3. 5 The proposed method with adaptive time step

5. The simulation in Ref. [6] uses adaptive time step with time step factor F=1.5, initial time step=10 ns and Newton iterations constraints n_{min}=5 and n_{max}=10 . The simulation results are shown in Table. 4.3.6. The simulation produced 62 samples and a total of 380 Newton iterations. The fundamental frequency was 5.035 MHz. The output voltage of the multi-time solution and the steady-state solutions are show in Fig. 4.3.17 and Fig. 4.3.18 respectively.

Simulation section	Number of solution samples along τ_2	Number of Newton iterations	Total Simulation time
First step	62	372	22.9 seconds
Regular HB	NA	8	

Table. 4.3. 6 Simulation results from Ref [6]

6. Also in this oscillator for some choices of the initial time step along τ_2 the method in Ref. [6] and the proposed method produce the unstable DC operating point solution. However, the proposed method finds the solution for some of these choices. For example the method in Ref. [6] fails to find the steady-state solution for initial time steps: 5 ns, 4 ns, 1 ns, 0.1 ns and 0.01 ns while the proposed method finds the steady-state solution for 5 ns and 4 ns and fails for 1 ns,0.1 ns and 0.01 ns.

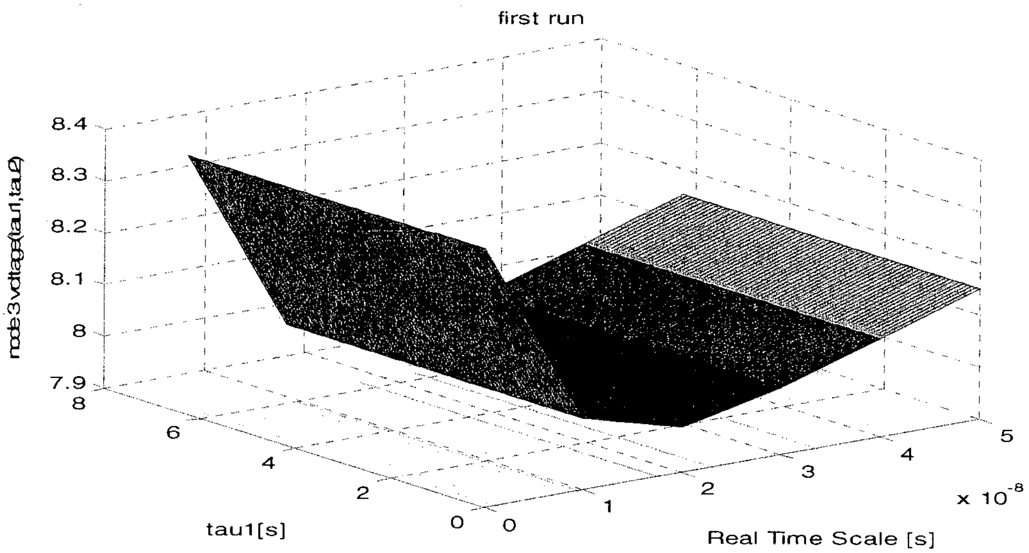


Fig.4.3. 10 The output voltage during first run

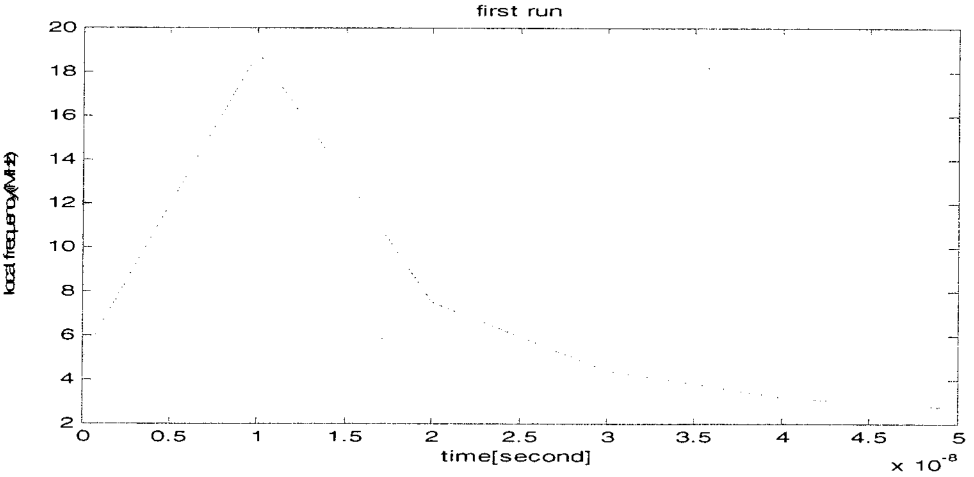


Fig.4.3. 11 Fundamental frequency during first run

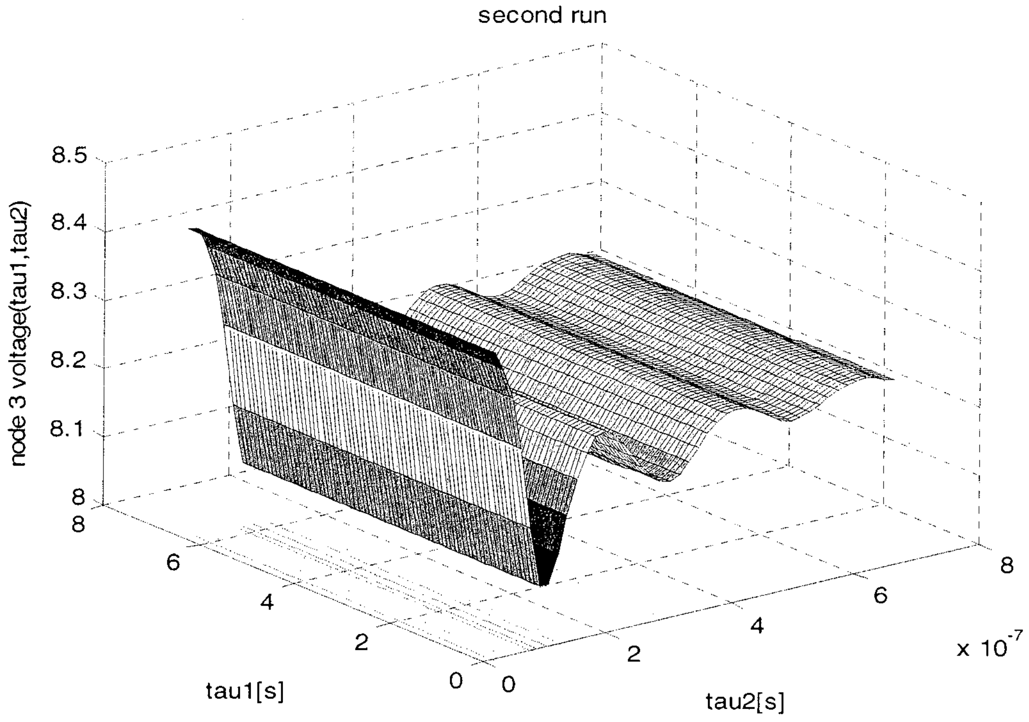


Fig.4.3. 12 The output voltage during the second run

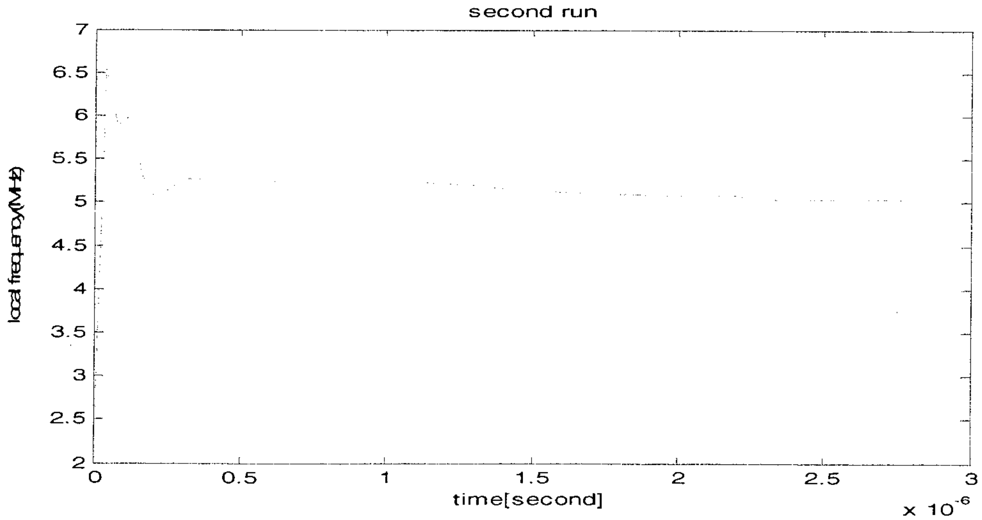


Fig.4.3. 13 Fundamental frequency during second run

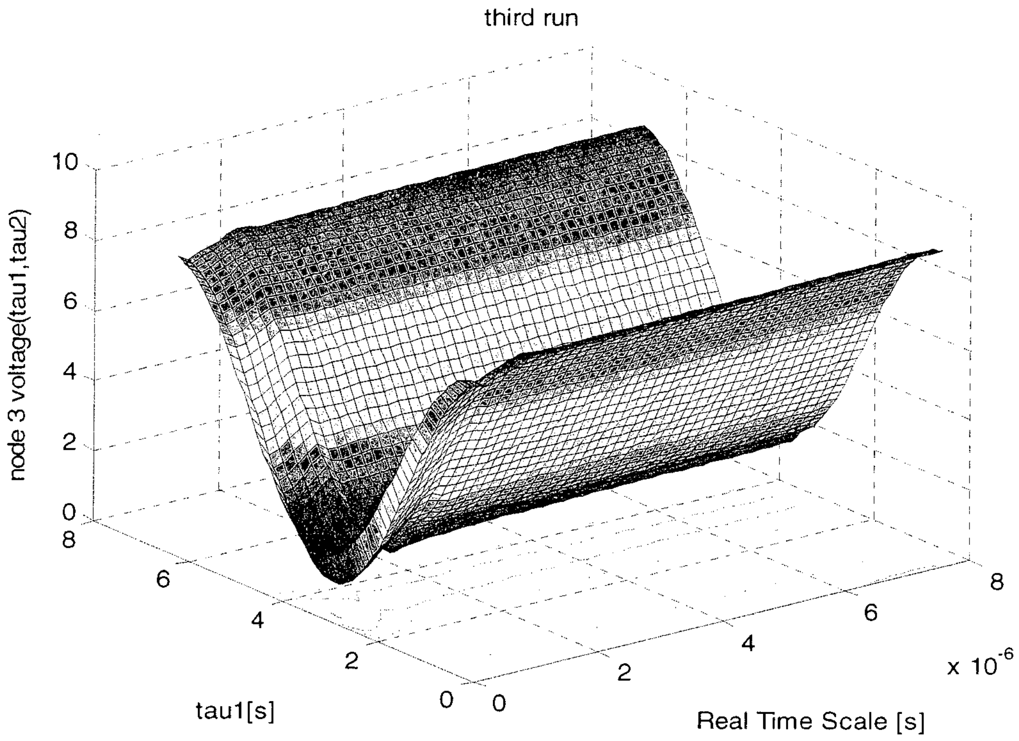


Fig.4.3. 14 The output voltage during third run

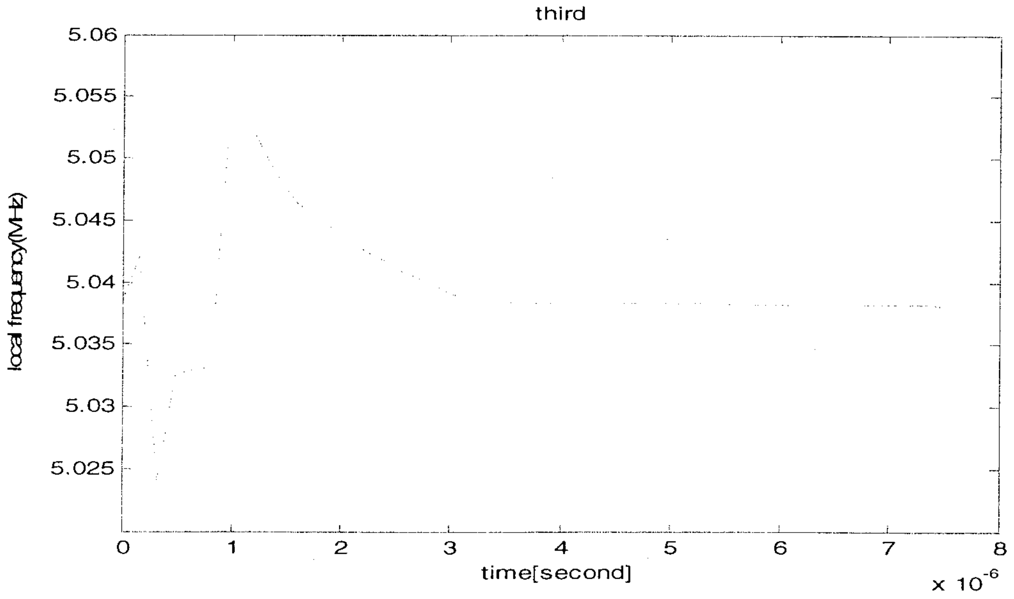


Fig.4.3. 15 The fundamental frequency during third run

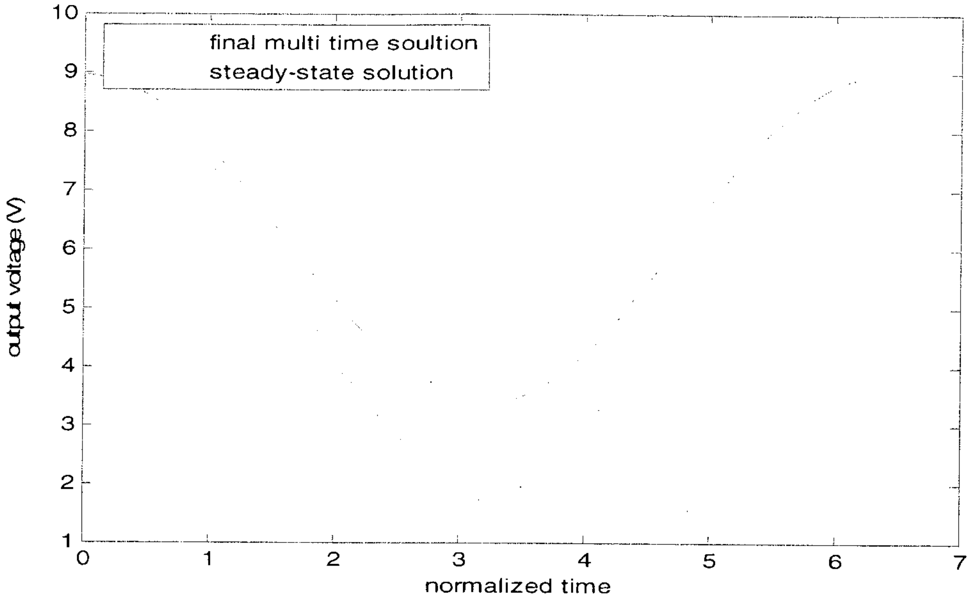


Fig.4.3. 16 Steady-state solution compared to final multi-time solution

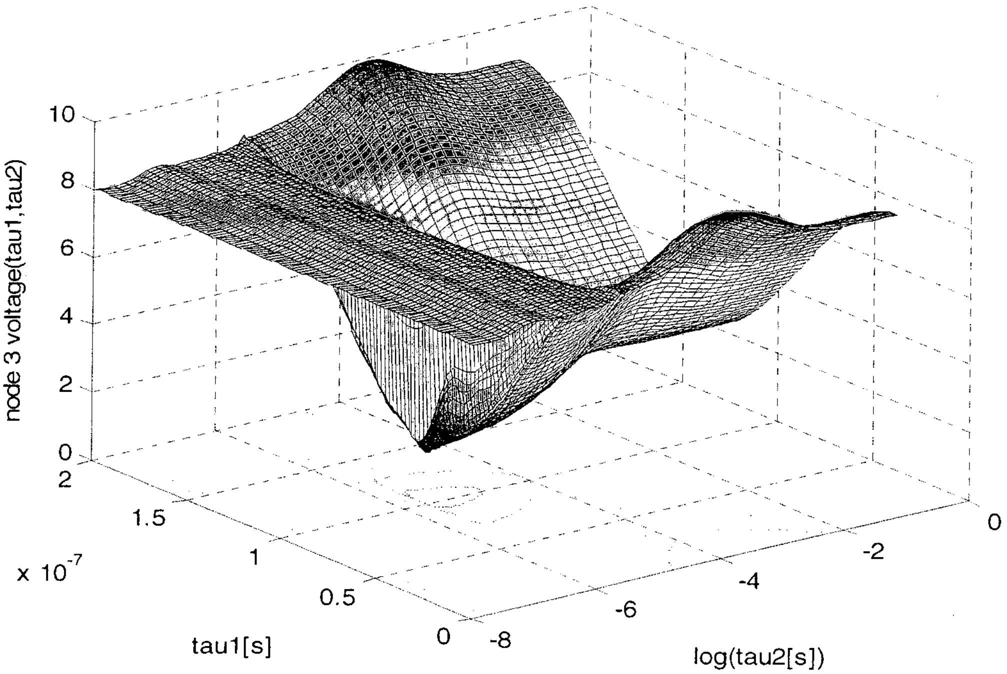


Fig.4.3. 17 The output voltage in Ref [6]

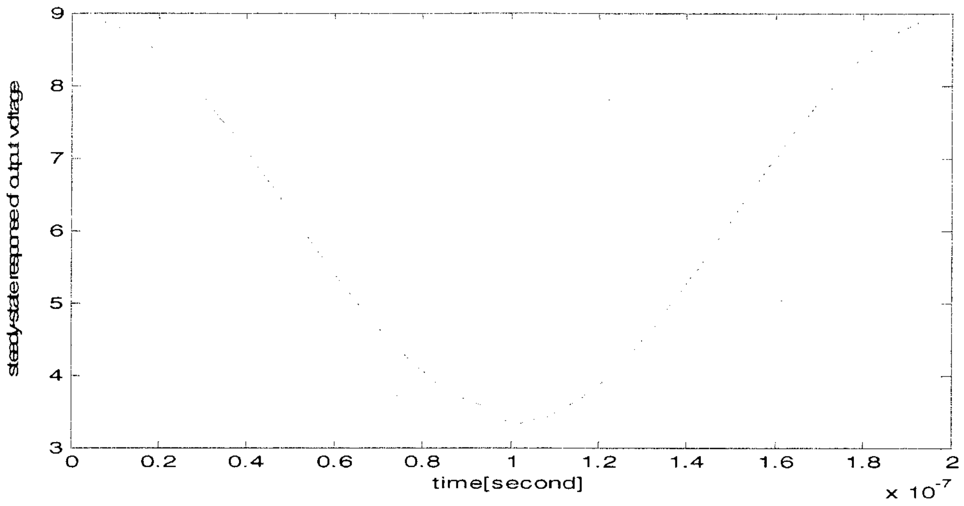


Fig.4.3. 18 Steady-state solution in Ref [6]

In this oscillator, the proposed method slightly improved the results in Ref. [6] in terms of reducing the number of Newton iterations needed to capture the steady-state, thus resulting in saving in the memory of the CPU. Using constant time step in Table. 4.3.1; the simulation needed 299 Newton iterations for both runs and 7 for the regular HB. While in [6] there are 372 Newton iterations and 8 for the regular HB. Here we have about 25% reduction in Newton iterations. The MPE is very fast, its simulation time is very small; its fraction of a second to extrapolate tenth of samples.

The use of adaptive time step with proposed method has not reduced the number of Newton iterations. Using three runs resulted in more Newton iterations than using two run but the final multi-time solution and the steady-state solution are very similar. In terms of simulation time, there is no considerable difference between the method in [6] and the proposed method; the average time saving is about 10%. The results in [6] which we compared our results to are the optimum results method [6] provides. Results in [6] uses adaptive time step factor=1.5, if this is set to 1.7; the solution produced is the unstable DC operating point solution, if it set to 1.4, it needs 400 Newton iterations and without adaptive time step (i.e. time step factor =1); it needs 45157 Newton iterations and takes 38 minutes. The simulation in method [6] might need to run for few times with modifying the adaptive time step factors to get the steady-state solution in a small number of Newton iteration. In this oscillator, the proposed method is as well sensitive to the simulation parameters. It is difficult to get the optimum results in the first try. However, the results in this oscillator might indicate the proposed method might be slightly more reliable than the method in Ref. [6]; it might not produce the optimum results but the right solution seems obtainable for wider range of simulation parameters.

Chapter 5

Conclusions and Future Research.

5.1 Conclusions

For the first time Minimal Polynomial Extrapolation (MPE) was used to accelerate capturing the oscillators' steady-state in frequency-domain. The method in Ref. [6] improves capturing the oscillator's steady-state by improving the initial guess of the regular Harmonic Balance using WaMPDE. The proposed method has further accelerated the work of WaMPDE to find the improved initial guess for the regular HB.

These improvements are measured by the reduction in Newton iterations and the simulation time. Number of Newton iterations in LC-tuned bipolar was reduced by 40% to 50% and for the Colpitts was reduced by around 20%. Significant time saving is not observed in these oscillators and this is not due to the work of MPE; MPE algorithm takes extremely little time (i.e. 0.001 seconds to extrapolate 4 samples of 10 length), we attribute this to: Firstly, the simulation time for these cases studies is relatively short; it is around 5 seconds and 20 seconds for the LC-tuned bipolar and Colpitts oscillators respectively. Secondly, the implemented code is reinitialized every time for each run (WaMPDE) and the regular HB. Time saving could be expected in systems which take longer simulation time. However, in these cases studies, the time saving might be accomplished if the program is written more efficiently. One simple optimization that could be implemented in the code is to avoid re-initializing all WaMPDE variables every time a run starts. The additional simulation time added by the work of MPE is very little time or rather insignificant to the total simulation time.

The limitation of the proposed method is that it is not straightforward to find the simulation parameters that result in optimum performance. The method can be more reliable if constant time step is used for all runs: the solution in Colpitts oscillator is not obtained if the time step of second run is increased by 6 times.

The few case studies presented here indicate that the proposed method is more efficient when oscillations can be started quickly as with the LC-tuned bipolar oscillator. In contrast, it is more difficult to start oscillations in the Colpitts oscillator and thus more steps are required to find a good initial guess for HB.

5.2 Future Research

We will suggest some ideas for further studies using the proposed method.

1. Improvements can be done on optimizing the simulation parameters. If the method is incorporated with a more reliable adaptive time step algorithm, this might turn out to be beneficial. Improvements can be done on studying the time step, in particular, the initial time step.
2. The method to start the oscillation should be reviewed and improved since for some initial choices of the time step along τ_2 the method fails to converge to the oscillatory steady-state solutions.
3. The method might be useful to determine the frequency locking range in forced oscillators in Ref. [31].
4. The proposed method should be tested with oscillators with more nodes to verify its effectiveness.

Appendix A

Newton-Raphson Method

The Newton-Raphson method is an iterative technique for solving nonlinear algebraic equations numerically. Like so much of the deferential calculus, it is based on the idea of linear approximation. [16,17].

Given the function:

$$f(x) = 0 \tag{A.1}$$

Iteration method:

Starts from initial value x^0

According to iteration function F: $x^{n+1} = F(x^n)$, generate a sequence of x^{n-1}, x^n , x^{n+1}, which hopefully converge to the solution x^*.

To solve the function in (A.1) with Newton-Raphson method: linearize the system and solve:

Taylor series $$f(x) = f(x^*) + \frac{d(x^*)}{dx}(x - x^*) \tag{A.2}$$

$$f(x^k) = f(x^k) + \frac{d(x^k)}{dx}(x^{k+1} - x^k) \tag{A.3}$$

Iteration function $$x^{k+1} = x^k - \left[\frac{df(x^k}{dx}\right]^{-1} f(x^k) \tag{A.4}$$

Each step, the function (*f*) and its derivatives is evaluated:

Do k=0,1,2...

until convergence

Given a system of N nonlinear equations:

$$F(x) = 0 \qquad (A.5)$$

System (A.5) will be solved by Newton-Raphson method as follows;

Taylor series $F(x) = F(x^*) + J(x^*)(x - x^*)$ (A.6)

Iteration function $x^{k+1} = x^k - J^{-1}F(x^k)$, (A.7)

where J is the Jacobian matrix and given by:

$$J(x) = \begin{bmatrix} \dfrac{\partial F_1(x)}{\partial x_1} & \cdots & \dfrac{\partial F_1(x)}{\partial x_N} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial F_N(x)}{\partial x_1} & \cdots & \dfrac{\partial F_N(x)}{\partial x_N} \end{bmatrix}$$

The Jacobian matrix is formulated by one of the numerical differentiation methods, depending on the desired accuracy.

Appendix B

Finite Difference Methods

Finite difference methods are techniques of numerical analysis to produce an estimate of the derivatives of a mathematical function. There are many finite difference methods, few of them are mentioned here. A method is selected depending on the desired accuracy. The methods can be expressed in Taylor to estimate the truncation error. Here are some examples of these methods using one dependant variable and expressing only the first derivative.

(1) A **forward difference**

$$\frac{df}{dx} = \frac{f(x+h)-f(x)}{h} + O(h)$$

$$\frac{df}{dx} \approx \frac{f(x+h)-f(x)}{h} \qquad \text{(B.1)}$$

the truncation error $O(h)$ is 1st order

(2) A **backward difference**

$$\frac{df}{dx} \approx \frac{f(x)-f(x-h)}{h} \qquad \text{(B.2)}$$

the truncation error $O(h)$ is 1st order

(3) Central Difference:

$$\frac{df}{dx} \approx \frac{f(x+h)-f(x-h)}{2h} \qquad \text{(B. 3)}$$

the truncation error$O(h^2)$ is 2nd order

(4) Five point formula

$$\frac{df}{dx} \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} \qquad (B.4)$$

the truncation error$O(h^4)$ is 4th order

These equations are used to approximate the numerical integration:

$$y'(t) = f\big(t, y(t)\big), \qquad y(t_0) = y_0$$

Using Equation (B.1), we can compute the solution by the following recursive scheme:

$$y_{n+1} = y_n + hf(t_n, y_n) \qquad (B.5)$$

This is the Euler method (or forward Euler method)

If Equation (B.2) is used, we have:

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}) \qquad (B.6)$$

This equation is known as Backward Euler rule

Bibliography

[1] O. Narayan and Jaijeet Roychowdhury, "Analyzing circuits with widely separated time scales using numerical PDE methods", *IEEE Transactions on Circuits and Systems—I: Fundamental Theory and Applications*, Vol. 48, No. 5, May 2001, pp. 578-594.

[2] H. Asai and H. Makino, "Frequency domain latency and relaxation-based harmonic analysis of nonlinear circuits," *Proceedings of the 34th Midwest Symposium on Circuits and Systems*, Vol. 1, May 1991, pp. 202–205.

[3] M. Gourary, S. G. Rusakov, S. L. Ulyanov, M. M. Zharov, K. K. Gullapalli and B. J Mulvaney, "A new technique to exploit frequency domain latency in harmonic balance simulators," *Proceedings of the 1999 Design Automation Conference*, Vol. 1, Jan. 1999, pp. 65–68.

[4] N. Soveiko and M. Nakhla," Wavelet harmonic balance," *IEEE Microwave and Wireless Components Letters*, Vol. 13, No. 6, June 2003, pp. 232-234.

[5] Lei Zhu and Carlos E. Christoffersen, " Adaptive harmonic balance analysis of oscillators using multiple time scales," *3rd International IEEE Northeast Workshop on Circuits and Systems Digest*, June 2005, pp. 187-190.

[6] Lei Zhu and Carlos E. Christoffersen, "Transient and steady-state analysis of nonlinear RF and microwave circuits," *EURASIP Journal on Wireless Communications and Networking*, Vol. 2006, No. 2, Special Issue on CMOS RF Circuits for Wireless Applications, Article ID 32097, pp. 1-11.

[7] O. Narayan and J. Roychowdhury, "Analyzing oscillators using multitime PDEs," *IEEE transactions on circuits and systems—I: fundamental theory and applications*, Vol. 50, No. 7, July 2003, pp. 894-903.

[8] D. Smith, W. Ford and A. Sidi, "Extrapolation methods for vector sequences," SIAM Review, Vol. 29(2), June 1987, pp. 199-233.

[9] S. Cabay and L.W. Jackson, " A Polynomial extrapolation method for finding limits and antilimits of vector sequences," *SIAM J. Numer Anal*, Vol 13, pp. 734-752.

[10] Jack R.Smith, *Modern Communication Circuit, second edition*,1998, WCB/McGraw-Hill.

[11] T. J. Aprille and T. N. Trick, "Steady-state analysis of nonlinear circuits with periodic inputs", *proceedings of the IEEE*, Vol. 60, No. 1, January 1972, pp. 108-114.

[12] Carlos E. Christoffersen, "State variable harmonic balance simulation of a quasi-optical power combining system," *Master's thesis, North Carolina State University, 1998.*

[13] V. Rizzoli, A. Constanzo and A.Neri, "Harmonic balance analysis of microwave oscillators with automatic suppression of degenerate solution," *Electronic Letters*, Vol. 28, No. 3, Jan 1992, pp. 256-257.

[14] A. Brambilla and P.Maffezzoni, "Envelope-following method to compute steady state solutions of electrical circuits," *IEEE transaction on circuits and systems-1: fundamental theory and applications,* Vol. 47, July 2000, pp. 999- 1008.

[15] Richard R. Spencer and Mohammed S.Ghusai, *Introduction to Electronics Circuit Design*, 2003, Prentice Hall.

[16] Jack A.Smith, *Modern Communication Circuits, second edition.* 1997 McGraw-Hill.

[17] L. Petzold, "An efficient numerical method for highly oscillatory ordinary differential equations," *SIAM J. Numer. ANAL* Vol. 18, No. 3, June 1981.

[18] T.Mei and J. Roychowdhury, "A robust envelope following method applicable to both non-autonomous and oscillatory circuits," *Annual ACM IEEE Design Automation Conference*, 2006, pp. 1029 - 1034.

[19] S. Skelboe, "Computation of the periodic steady-state response of nonlinear networks by extrapolation methods," *IEEE Trans. Circuits Syst.*, Vol. CAS-27, No. 3, March 1980, pp. 161- 175.

[20] M.S. Nakhla and J. Vlach, "piecewise harmonic balance technique for determination of periodic response of nonlinear systems," *IEEE Trans. on Circuit and Systems,* CAS-23, Feb 1976, pp. 85- 91.

[21] R. Gilmore, "Nonlinear circuit design using the modified harmonic balance algorithm," *IEEE Trans. On Microwave Theory and Techniques*, Vol. MTT-34, No. 12, Dec 1986, pp. 1294- 1307.

[22] Jorge F. Oliveira and José Carlos Pedro, "An efficient time-domain simulation method for multirate RF nonlinear Circuits," *IEEE Transactions on Microwave Theory and techniques*, Vol. 55, No. 11, November 2007, pp. 2384- 2392.

[23] Narayan and J. Roychowdhury, "Analysing forced oscillators with multiple time Scales,"*12th International Conference on VLSI Design,* January I999, pp. 621-624.

[24] Roland Pulch, "Finite difference methods for multi time scale differential algebraic equations," *ZAMM · Z. Angew. Math. Mech.* 83, No. 9, August 2003, pp. 571– 583.

[25] A. Semlyen and A. Medina, "Computation of the periodic steady state in systems with nonlinear components using a hybrid time and frequency domain methodology," *IEEE Transactions on Power Systems*, Vol. 10, No. 3, August 1995, pp. 637- 643.

[26] A. Brambilla, D. D'Amore and M. Santomauro, "Simulation of autonomous circuits in the time domain," *In ECCTD'95 European Conference on Circuit Theory & Design,* 1995, pp. 399- 402.

[27] Houben, "The numerical simulation of electrical oscillators," *The Netherlands, Printed by Eindhoven University Press, "* 2003.

[28] J.R.Parkhurst and L.L.Ogborn, "Determining the steady-state output of nonlinear oscillatory circuits using Multiple Shooting," *IEEE Trans. CAD*, Vol. 14, No. 7, Jul 1995, pp. 882- 889.

[29] J. Houben , W. Ter Maten , J. Maubach and J.M.F. Peters, " Novel time-domain methods for free-running oscillators," *Proceedings of the 15th European Conference Circuit Theory and Design, Finland"*, 2001, pp. 393- 396.

[30] A.Suarez and R.Quere, *Stability analysis of nonlinear microwave circuits,* 2003 Artech House, INC.

[31] Carlos. E. Christoffersen, M. Condon and T. Xu, "A new method for the determination of the locking range of oscillators," *Proceedings of the 2007 European Conference on Circuit Theory and Design (ECCTD07),* Seville, August 2007, pp. 575-578.