

City University of Hong Kong

Department of Mathematics

Bachelor of Science (Hons) in Computing Mathematics

Final Year Project Report

**TITLE: The Method of Multiple Scales and the
Perturbation-Incremental Method for Autonomous Non-linear
Oscillators**

Student Name: Ng Kwok Tim

Student Number:

Supervisor: Dr. K. W. Chung

Date: 27th March 2008

Table of Contents

1. Acknowledgement	p. 3
2. Abstract	p. 4
3. Introduction	p. 5 – p. 7
4. Methods	p. 8 – p. 17
- 4.1 Method of Multiple Scales	
- 4.2 Perturbation-Incremental Method	
5. Stability of Periodic Solution	p. 18 – p. 19
6. Result	p. 20 – p. 38
7. Discussion	p. 39 – p. 40
8. Conclusion	p. 41
9. Reference	p. 42
10. Appendix	p. 43 – p.56

1. Acknowledgement

I would like to thank my supervisor, Dr K. W. Chung. He gave a lot of ideas and comments for this project. Besides, he taught me a lot of things that I have not learnt in my previous courses. For example, he taught me how to use in a different approach to solve problems using Matlab. Also, before taking this project, I had neither taken any course on elementary numerical method nor learnt any programming software such as Matlab and Maple. Dr Chung gave me ideas on how to use analytic and numerical methods to compute limit cycles accurately. Even though I could get my result in a stupid way by work out all the things explicitly using Matlab, he taught me a systemic and efficient way in writing the programs. That is why I am proud to be under his supervision. Under his professional guidance and supervision, I obtain interesting results for my final year project.

In addition, I would like to thank Dr K. W. Chung again to give me a precious chance in taking this project and I have learnt a lot from him. This project is definitely a valuable experience in my life.

2. Abstract

In this report, a method of multiple scales is presented for the analysis of weakly non-linear oscillators in the form $\frac{d^2u}{dt^2} + \omega_0^2 u = \varepsilon F\left(u, \frac{du}{dt}\right)$, where $F\left(u, \frac{du}{dt}\right)$ is arbitrary non-linear function of its arguments and $0 < \varepsilon \ll 1$. Maple, a computer tool, is used for the analysis.

Then a perturbation-incremental method is presented for the analysis of strongly non-linear oscillators in the form $\ddot{x} + g(x) = \lambda f(x, \dot{x})\dot{x}$, where $g(x)$ and $f(x, \dot{x})$ are arbitrary non-linear functions of their arguments and $\lambda > 0$. The perturbation-incremental method is an extension of the classical perturbation method to the case where λ is not necessarily small. The method incorporates salient features from both the perturbation method and the incremental method. Matlab is used for the perturbation-incremental method. Limit cycles of an oscillator can be calculated to any desired degree of accuracy. The stability of a limit cycle is also considered.

3. Introduction

The study of both weakly and strongly non-linear oscillators has been developed for quite a long period of time. Various perturbation methods and numerical integration methods such as the method of multiple scales [1, 2] and Runge-Kutta method have been applied in the analysis of the classical non-linear oscillators. The classical perturbation methods such as the method of multiple scales are restricted to solve weakly non-linear oscillators with the governing equation

$$\ddot{u} + \omega_0^2 u = \varepsilon F(u, \dot{u}) \quad (3.1)$$

where $0 < \varepsilon \ll 1$, ω_0 is a given constant and $F(u, \dot{u})$ is arbitrary non-linear function of its arguments. Traditionally, the perturbation methods can obtain approximate analytical solutions. However, the restriction of these methods is that the parameter ε must be very small (i.e. $0 < \varepsilon \ll 1$). That is why the method of multiple scales or the classical perturbation methods is not useful whenever $\varepsilon \geq 1$. Actually, the method of multiple scales is not suitable to solve the strongly non-linear oscillators of the following form

$$\ddot{x} + g(x) = \lambda f(x, \dot{x}) \quad (3.2)$$

where $\lambda > 0$ is an arbitrary parameter and $g(x)$, $f(x, \dot{x})$ are arbitrary non-linear functions of their arguments. We will apply the method of multiple scales to obtain analytical solutions and then compare with those obtained by the Runge-Kutta method.

The numerical integration methods such as the Runge-Kutta method are capable of solving such strongly non-linear oscillators. However, the existence, uniqueness and stability of the solution should be ensured in order to apply a numerical integration method. Also, the iterative scheme of a numerical integration requires suitable initial conditions to be given. More importantly, the numerical integration methods are not suitable for obtaining unstable solutions. Owing to these kinds of difficulties with both the classical perturbation method and the numerical integration methods, a new method (Perturbation-Incremental method) is developed in order to overcome those difficulties occurred in the old methods.

The Perturbation-Incremental method [4], as its name suggests, basically involves the following two steps: a first perturbation step and a second parameter incremental step. Here is a brief introduction. Firstly, we introduce a non-linear time transformation and rewrite the transformed differential equation as an integral equation. Then, a perturbation method is used to obtain the initial solution for the case $\lambda \approx 0$. Once we obtain the initial solution, we use an incremental method to obtain an approximate periodic orbit for arbitrary large value of a parameter λ and finally determine the stable and/or unstable periodic solution. Therefore, the Perturbation-Incremental method is a combination of the perturbation and numerical iterative methods which can obtain the solutions of the strongly non-linear oscillator. The numerical results from the Perturbation-Incremental method

which are calculated using Matlab [5] will be compared with those obtained from the Runge-Kutta method. The stability of the limit cycle can be determined by the Floquet method [6, 7]. We calculate the characteristic (Floquet) exponent directly from the solution to analyze the stability of a limit cycle.

4. Methods

In this report, we introduce two methods. The first one is the Method of Multiple Scales, and the other one is the Perturbation-Incremental Method.

4.1. Method of Multiple Scales

In order to illustrate the method of multiple scales, we will consider its application to weakly nonlinear oscillators with the following differential equation

$$\frac{d^2u}{dt^2} + \omega_0^2 u = \varepsilon F\left(u, \frac{du}{dt}\right), \quad (4.1.1)$$

where $\varepsilon \ll 1$, ω_0 is a given constant and F can be arbitrary non-linear function of its arguments. The main idea for this method is to introduce the time scales $T_0, T_1, \dots, T_M$ with

$$T_m = \varepsilon^m t, \quad (4.1.2)$$

where $m = 0, \dots, M$ and M is a positive integer.

Since we have introduced the $M+1$ time scales, we also assume $u(t; \varepsilon) = u(T_0, T_1, \dots, T_M; \varepsilon)$.

Thus we are now looking for a solution of the following form

$$u(t; \varepsilon) = \sum_{m=0}^M \varepsilon^m u_m(T_0, T_1, \dots, T_M) + O(\varepsilon^{M+1}), \quad (4.1.3)$$

where $O(\varepsilon^{M+1})$ is the error for the finite series of $u(t; \varepsilon)$. By using the time scales and the

chain rule, the time derivative is substituted by the following form

$$\frac{d}{dt} = \frac{\partial}{\partial T_0} + \varepsilon \frac{\partial}{\partial T_1} + \varepsilon^2 \frac{\partial}{\partial T_2} + \dots + \varepsilon^M \frac{\partial}{\partial T_M}. \quad (4.1.4)$$

Then (4.1.1) can be transform into $M + 1$ equation according to the orders in ε . Equations

(4.1.2) through (4.1.4) formulate the whole idea of the method of multiple scales. In

particular, we will only focus on three time scales ($M = 2$) and we will use the van der Pol's equation with $F = (1 - u^2) \left(\frac{du}{dt} \right)$ and $\omega_0 = 1$ as an illustration. Now, consider the van der Pol oscillator

$$\frac{d^2 u}{dt^2} + u = \varepsilon(1 - u^2) \left(\frac{du}{dt} \right), \quad u(0) = a_0, \quad \frac{du}{dt}(0) = 0. \quad (4.1.5)$$

With the time scales $T_0 = t$, $T_1 = \varepsilon t$ and $T_2 = \varepsilon^2 t$, the time derivatives can be written as

$$\frac{d}{dt} = \frac{\partial}{\partial T_0} + \varepsilon \frac{\partial}{\partial T_1} + \varepsilon^2 \frac{\partial}{\partial T_2}, \quad (4.1.6)$$

$$\frac{d^2}{dt^2} = \frac{\partial^2}{\partial T_0^2} + 2\varepsilon \frac{\partial^2}{\partial T_0 \partial T_1} + 2\varepsilon^2 \frac{\partial^2}{\partial T_0 \partial T_2} + \varepsilon^2 \frac{\partial^2}{\partial T_1^2} + 2\varepsilon^3 \frac{\partial^2}{\partial T_1 \partial T_2} + \varepsilon^4 \frac{\partial^2}{\partial T_2^2}. \quad (4.1.7)$$

We are now looking for a solution of the following form

$$u(t; \varepsilon) = u(T_0, T_1, T_2; \varepsilon) = u_0(T_0, T_1, T_2) + \varepsilon u_1(T_0, T_1, T_2) + \varepsilon^2 u_2(T_0, T_1, T_2) + O(\varepsilon^3). \quad (4.1.8)$$

Substituting equations (4.1.6) through (4.1.8) into (4.1.5) and equating the coefficients of like power of ε (i.e. $\varepsilon^0, \varepsilon^1$ and ε^2), we obtain 3 ordinary differential equations which can be solved one after the other:

$$\frac{\partial^2 u_0}{\partial T_0^2} + u_0 = 0, \quad (4.1.9)$$

$$\frac{\partial^2 u_1}{\partial T_0^2} + u_1 = (1 - u_0^2) \frac{\partial u_0}{\partial T_0} - 2 \frac{\partial^2 u_0}{\partial T_0 \partial T_1}, \quad (4.1.10)$$

$$\frac{\partial^2 u_2}{\partial T_0^2} + u_2 = (1 - u_0^2) \frac{\partial u_1}{\partial T_0} + (1 - u_0^2) \frac{\partial u_0}{\partial T_1} - 2u_0 u_1 \frac{\partial u_0}{\partial T_0} - 2 \frac{\partial^2 u_1}{\partial T_0 \partial T_1} - \frac{\partial^2 u_0}{\partial T_1^2} - 2 \frac{\partial^2 u_0}{\partial T_0 \partial T_2}. \quad (4.1.11)$$

The solution of equation (4.1.9) can be always written as the following form:

$$u_0 = A(T_1, T_2) e^{iT_0} + \bar{A}(T_1, T_2) e^{-iT_0}, \quad (4.1.12)$$

where $\bar{A}$ is a complex conjugate of A .

Substituting (4.1.12) into (4.1.10), we obtain the following ordinary differential equation:

$$\frac{\partial^2 u_1}{\partial T_0^2} + u_1 = -i(-A + 2\frac{\partial A}{\partial T_1} + A^2 \bar{A})e^{iT_0} - iA^3 e^{3iT_0} + i(-\bar{A} + 2\frac{\partial \bar{A}}{\partial T_1} + A\bar{A}^2)e^{-iT_0} + i\bar{A}^3 e^{-3iT_0}. \quad (4.1.13)$$

Thus, to avoid secular terms, we require the vanishing of the coefficients of e^{iT_0} and e^{-iT_0} :

$$-A + 2\frac{\partial A}{\partial T_1} + A^2 \bar{A} = 0. \quad (4.1.14)$$

Using (4.1.14), the solution of equation (4.1.13) can also be written as the following form:

$$u_1 = B(T_1, T_2)e^{iT_0} + \frac{1}{8}iA^3 e^{3iT_0} + \bar{B}(T_1, T_2)e^{-iT_0} - \frac{1}{8}i\bar{A}^3 e^{-3iT_0}. \quad (4.1.15)$$

To determine u_0 , we can let $A = \frac{1}{2}a(T_1, T_2)e^{i\phi(T_1, T_2)}$ into (4.1.14) and then separate the real

and the imaginary parts, we obtain the following 2 equations:

$$\frac{\partial \phi}{\partial T_1} = 0, \quad \frac{\partial a}{\partial T_1} = \frac{1}{2}(1 - \frac{1}{4}a^2)a. \quad (4.1.16)$$

Thus we get

$$\phi = \phi(T_2), \quad a = \sqrt{\frac{4}{1 + c(T_2)e^{-T_1}}}. \quad (4.1.17)$$

Similarly, we can get u_1 by substituting (4.1.15) into the equation (4.1.11) and then

vanish the coefficients of e^{iT_0} and e^{-iT_0} in order to avoid the secular terms. Here, for

simplicity, we will only consider the first approximation to u by letting the B , ϕ and c as

constants. Then, from the initial conditions $u(0) = a_0$ and $\frac{du}{dt}(0) = 0$, we get

$$u = \frac{2}{\sqrt{1 + (\frac{4}{a_0^2} - 1)e^{-\varepsilon t}}} \cos t + O(\varepsilon). \quad (4.1.18)$$

Thus, we see from solution (4.1.18) that the expansion tends to the limit cycle

$u = 2 \cos t + O(\varepsilon)$ for all initial values as $t \rightarrow \infty$. In particular, we also find that the solution

(4.1.18) is periodic if and only if the initial value $a_0 = 2$. The second approximation to u

for the method of multiple scales is given in section 6 which is done using the software MAPLE. The Maple program for this method is attached in the CD.

4.2. Perturbation-Incremental Method

For the Perturbation-Incremental Method, we will consider the strongly non-linear oscillators of the form listed below

$$\ddot{x} + g(x) = \lambda f(x, \dot{x})\dot{x}, \quad (4.2.1)$$

where $\lambda > 0$ is an arbitrary parameter and $g(x)$, $f(x, \dot{x})$ are arbitrary non-linear functions of their arguments. First, we introduce a time transformation in the following form

$$\frac{d\varphi}{dt} = \Phi(\varphi), \quad \Phi(\varphi + 2\pi) = \Phi(\varphi), \quad (4.2.2)$$

where φ is the new time and Φ is a periodic function with 2π period. Here, we assume that the equation (4.2.1) possesses at least one limit cycle solution. And we also assume the origin of the $x-\dot{x}$ phase plane is an interior point of the limit cycle. Then, in the φ domain the limit cycle can be written as the following

$$x = a \cos \varphi + b, \quad a > 0, \quad (4.2.3)$$

where a is the amplitude, b the bias and φ is between 0 to 2π .

Using (4.2.2) and (4.2.3), equation (4.2.1) can be written into the φ domain,

$$\Phi \frac{d}{d\varphi} (\Phi x') + g(x) = \lambda f(x, \Phi x') \Phi x', \quad (4.2.4)$$

where primes denote differentiation with respect to φ . Multiplying both sides of equation

(4.2.4) by $x' = -a \sin \varphi$ and then follow by integration, we obtain the following

$$\frac{1}{2}(\Phi \sin \varphi)^2 + \tilde{v}(a, b, \varphi) - \lambda \int_0^\varphi \tilde{f}(a, b, \Phi, \theta) d\theta = 0. \quad (4.2.5)$$

For convenience, we have already introduced the following notation

$$\tilde{v}(a, b, \varphi) = \frac{v(a \cos \varphi + b) - v(a + b)}{a^2}, \quad (4.2.6)$$

and

$$\tilde{f}(a, b, \Phi, \theta) = f(a \cos \theta + b, -a\Phi(\theta) \sin \theta) \Phi(\theta) \sin^2 \theta, \quad (4.2.7)$$

where

$$v(x) = \int_0^x g(u) du. \quad (4.2.8)$$

By taking $\varphi = \pi$ and $\varphi = 2\pi$ in equation (4.2.5), we also obtain the following

$$\tilde{v}(a, b, \pi) - \lambda \int_0^\pi \tilde{f}(a, b, \Phi, \theta) d\theta = 0, \quad (4.2.9)$$

$$\int_0^{2\pi} \tilde{f}(a, b, \Phi, \theta) d\theta = 0. \quad (4.2.10)$$

First step: perturbation method ($\lambda \approx 0$)

For $\lambda \approx 0$, assume the solution of equations (4.2.5), (4.2.9) and (4.2.10) can be represented

in the form

$$a = a_0 + O(\lambda), \quad b = b_0 + O(\lambda), \quad \Phi = \Phi_0 + O(\lambda). \quad (4.2.11)$$

Then we obtain

$$\Phi_0(\varphi) = \frac{\sqrt{-2\tilde{v}(a_0, b_0, \varphi)}}{|\sin \varphi|}, \quad (4.2.12)$$

where the constants a_0 and b_0 can be found from the following equations

$$v(-a_0 + b_0) - v(a_0 + b_0) = 0, \quad (4.2.13)$$

$$\int_0^{2\pi} \tilde{f}(a_0, b_0, \Phi_0, \theta) d\theta = 0. \quad (4.2.14)$$

To solve equation (4.2.14), we usually use the numerical integration method such as the Simpson's Rule. After we obtain the constants a_0 , b_0 and the function $\Phi_0(\varphi)$, the zero-order perturbation solution for the limit cycle of equation (4.2.1) can be determined:

$$x = a_0 \cos \varphi + b_0, \quad \frac{dx}{dt} = -a_0 \Phi_0(\varphi) \sin \varphi. \quad (4.2.15)$$

Thus, the first step of the perturbation-incremental method provides an initial solution for the second step which is an iterative process.

Second step: parameter incremental method—a Newton-Raphson procedure ($\lambda = \lambda_0 + \Delta\lambda$)

Small increments are added to the current solution a_0 , b_0 and Φ_0 (where a_0 , b_0 , Φ_0 are the perturbation solution from the first step when $\lambda_0 = 0$) of equations (4.2.5), (4.2.9) and (4.2.10) in order to obtain a neighboring solution corresponding to $\lambda = \lambda_0 + \Delta\lambda$ ($0 < \Delta\lambda \ll 1$) and

$$a = a_0 + \Delta a, \quad b = b_0 + \Delta b, \quad \Phi = \Phi_0 + \Delta\Phi. \quad (4.2.16)$$

Once we solve for Δa , Δb and $\Delta\Phi$ from (4.2.16), we get the neighboring solution corresponding to $\lambda = \lambda_0 + \Delta\lambda$. To do this, we expand (4.2.5), (4.2.9) and (4.2.10) in Taylor's series about the initial state. The linearized incremental equations are obtained by ignoring all the nonlinear terms in the small increments.

From (4.2.5)

$$\begin{aligned}
& \left[\left(\frac{\partial \tilde{v}(a, b, \varphi)}{\partial a} \right)_0 - \lambda \int_0^\varphi \left(\frac{\partial \tilde{f}}{\partial a} \right)_0 d\theta \right] \Delta a + \left[\left(\frac{\partial \tilde{v}(a, b, \varphi)}{\partial b} \right)_0 - \lambda \int_0^\varphi \left(\frac{\partial \tilde{f}}{\partial b} \right)_0 d\theta \right] \Delta b \\
& + (\Phi_0 \sin^2 \varphi) \Delta \Phi - \lambda \int_0^\varphi \left(\frac{\partial \tilde{f}}{\partial \Phi} \right)_0 \Delta \Phi d\theta \\
& = -\frac{1}{2} (\Phi_0 \sin \varphi)^2 - \tilde{v}(a_0, b_0, \varphi) + \lambda \int_0^\varphi \tilde{f}(a_0, b_0, \Phi_0, \theta) d\theta. \quad (4.2.17)
\end{aligned}$$

From (4.2.9)

$$\begin{aligned}
& \left[\left(\frac{\partial \tilde{v}(a, b, \pi)}{\partial a} \right)_0 - \lambda \int_0^\pi \left(\frac{\partial \tilde{f}}{\partial a} \right)_0 d\theta \right] \Delta a + \left[\left(\frac{\partial \tilde{v}(a, b, \pi)}{\partial b} \right)_0 - \lambda \int_0^\pi \left(\frac{\partial \tilde{f}}{\partial b} \right)_0 d\theta \right] \Delta b - \lambda \int_0^\pi \left(\frac{\partial \tilde{f}}{\partial \Phi} \right)_0 \Delta \Phi d\theta \\
& = -\tilde{v}(a_0, b_0, \pi) + \lambda \int_0^\pi \tilde{f}(a_0, b_0, \Phi_0, \theta) d\theta. \quad (4.2.18)
\end{aligned}$$

From (4.2.10)

$$\Delta a \int_0^{2\pi} \left(\frac{\partial \tilde{f}}{\partial a} \right)_0 d\theta + \Delta b \int_0^{2\pi} \left(\frac{\partial \tilde{f}}{\partial b} \right)_0 d\theta + \int_0^{2\pi} \left(\frac{\partial \tilde{f}}{\partial \Phi} \right)_0 \Delta \Phi d\theta = - \int_0^{2\pi} \tilde{f}(a_0, b_0, \Phi_0, \theta) d\theta, \quad (4.2.19)$$

where

$$\left(\frac{\partial \tilde{v}(a, b, \varphi)}{\partial a} \right)_0 = \frac{\partial}{\partial a} \tilde{v}(a, b, \varphi) \Big|_{a=a_0, b=b_0}, \quad \left(\frac{\partial \tilde{f}}{\partial a} \right)_0 = \frac{\partial}{\partial a} \tilde{f}(a, b, \Phi, \theta) \Big|_{a=a_0, b=b_0, \Phi=\Phi_0},$$

and the other terms are defined similarly.

Since Φ is a periodic function with 2π period, and we know that any 2π periodic function can be represented by a Fourier expansion. Here, we have a basic assumption that M harmonics will provide a sufficiently accurate representation for our solution. Thus, we use the Fourier expansion to represent the periodic function Φ ,

$$\Phi_0 = \sum_{j=0}^M (p_j \cos j\varphi + q_j \sin j\varphi), \quad q_0 = 0. \quad (4.2.20)$$

The unknown $\Delta\Phi$ can also be expressed in that form

$$\Delta\Phi = \sum_{j=0}^M (\Delta p_j \cos j\varphi + \Delta q_j \sin j\varphi), \quad \Delta q_0 = 0. \quad (4.2.21)$$

We also expand the periodic functions in equations (4.2.17)-(4.2.19) into its Fourier series:

$$\tilde{v}(a, b, \varphi) = \sum_{k \geq 0} \tau_k \cos k\varphi, \quad (4.2.22)$$

$$\left(\frac{\partial \tilde{v}}{\partial a} \right)_0 = \sum_{k \geq 0} \alpha_k \cos k\varphi, \quad (4.2.23)$$

$$\left(\frac{\partial \tilde{v}}{\partial b} \right)_0 = \sum_{k \geq 0} \beta_k \cos k\varphi. \quad (4.2.24)$$

Here, the sine terms do not have any contribution because of (4.2.13). Also

$$\tilde{f}(a_0, b_0, \Phi_0, \varphi) = \sum_{k \geq 0} (\gamma_{1,k} \cos k\varphi + \delta_{1,k} \sin k\varphi), \quad \delta_{1,0} = 0, \quad (4.2.25)$$

$$\left(\frac{\partial \tilde{f}}{\partial a} \right)_0 = \sum_{k \geq 0} (\gamma_{2,k} \cos k\varphi + \delta_{2,k} \sin k\varphi), \quad \delta_{2,0} = 0, \quad (4.2.26)$$

$$\left(\frac{\partial \tilde{f}}{\partial b} \right)_0 = \sum_{k \geq 0} (\gamma_{3,k} \cos k\varphi + \delta_{3,k} \sin k\varphi), \quad \delta_{3,0} = 0, \quad (4.2.27)$$

$$\left(\frac{\partial \tilde{f}}{\partial \Phi} \right)_0 = \sum_{k \geq 0} (\gamma_{4,k} \cos k\varphi + \delta_{4,k} \sin k\varphi), \quad \delta_{4,0} = 0, \quad (4.2.28)$$

$$\Phi_0 \sin^2 \varphi = \sum_{k \geq 0} (\zeta_{1,k} \cos k\varphi + \eta_{1,k} \sin k\varphi), \quad \eta_{1,0} = 0, \quad (4.2.29)$$

$$\frac{1}{2} (\Phi_0 \sin \varphi)^2 + \tilde{v}(a_0, b_0, \varphi) = \sum_{k \geq 0} (\zeta_{2,k} \cos k\varphi + \eta_{2,k} \sin k\varphi), \quad \eta_{2,0} = 0. \quad (4.2.30)$$

Thus, by substituting the above Fourier series expansions into equations (4.2.17)-(4.2.19)

and applying the harmonic balance method, we obtain a system of $2M + 3$ linear equations

with $2M + 3$ unknowns (i.e. $\Delta a, \Delta b, \Delta p_0, \Delta p_j, \Delta q_j$, for $j = 1, \dots, M$) in the following form

$$A_n \Delta a + B_n \Delta b + A_{n,0} \Delta p_0 + \sum_{j=1}^M (A_{n,j} \Delta p_j + B_{n,j} \Delta q_j) = R_n, \quad (4.2.31)$$

where $n = 0, 1, 2, \dots, 2M + 2$, and the coefficients $A_n, B_n, A_{n,j}, B_{n,j}$ and R_n are given in the Appendix. We note that $A_i, B_i, A_{i,j}, B_{i,j}$ and R_i come from the coefficients of cosine terms ($\cos i\varphi$) while $A_{M+i}, B_{M+i}, A_{M+i,j}, B_{M+i,j}$ and R_{M+i} come from the coefficients of sine terms ($\sin i\varphi$) in the Fourier expansion of (4.2.17) (for $i = 1, \dots, M$). Further, $A_0, B_0, A_{0,j}, B_{0,j}$ and R_0 come from the constant term of Fourier expansion of (4.2.17). Finally, $A_{2M+1}, B_{2M+1}, A_{2M+1,j}, B_{2M+1,j}, R_{2M+1}$ are values from (4.2.18) and $A_{2M+2}, B_{2M+2}, A_{2M+2,j}, B_{2M+2,j}, R_{2M+2}$ from (4.2.19). Also, it should be noted that R_n in equations (4.2.31) are residue terms to prevent the incremental process drifting away from the actual solution (or to avoid the error going exponential fast). Here is our iterative scheme: equations (4.2.31) are to be solved by an equation solver such as the Gaussian elimination procedure. The values a_0, b_0 and Φ_0 are updated by adding together the original values and the corresponding incremental values. The iteration process continues until $R_n \rightarrow 0$ for all n (in practice, $|R_n|$ is less than a desired degree of accuracy). Here, in our report, the iteration process continues until the norm of R_n is less than 10^{-8} . The entire incremental process proceeds by adding the increment $\Delta\lambda$ (the control increment) to the converged value of λ , using the previous solution as the initial approximation until a new converged solution is obtained. Once we obtain a, b and Φ , we immediately determine the solution of the limit cycle as follows

$$x = a \cos \varphi + b, \quad \frac{dx}{dt} = -a\Phi(\varphi) \sin \varphi, \quad (4.2.32)$$

where φ varies from 0 to 2π .

5. Stability of Periodic Solution

A limit cycle is unstable, if there is a small derivation from the original path of limit cycle, and then the solution will leave the path of the limit cycle and never come back. The stability of a limit cycle can be determined using the Floquet method by calculating the characteristic exponent ρ directly from the solutions. We first rewrite equation (4.2.1) in the following form

$$\begin{aligned}\dot{x} &= y, \\ \dot{y} &= \lambda f(x, y)y - g(x).\end{aligned}\tag{5.1}$$

Let P be the Jacobain matrix of (5.1) as

$$P(t) = \begin{pmatrix} 0 & 1 \\ \lambda y \frac{\partial f}{\partial x} - \frac{dg}{dx} & \lambda(f(x, y) + y \frac{\partial f}{\partial y}) \end{pmatrix},\tag{5.2}$$

where P is a 2×2 matrix with period T and $x = x(t)$.

Let $\xi(t)$ represent a perturbation or disturbance of the original solution which means that it will affect the stability of the solution and satisfy the following system

$$\dot{\xi} = P(t)\xi,\tag{5.3}$$

Let μ and ρ be respectively the characteristic number and the characteristic exponent of system (5.3). Then, the product of the two characteristic numbers is given as

$$\mu_1 \mu_2 = \exp \int_0^T \text{tr} \{P(t)\} dt,\tag{5.4}$$

where $\text{tr} \{P(t)\}$ is the trace of $P(t)$ (the sum of the elements of its principal diagonal).

Now, the characteristic number μ is related to the characteristic exponent ρ as follow

$$\mu = e^{\rho T} . \quad (5.5)$$

It can be shown that one of the characteristic numbers in equation (5.4) is equal to one.

Then, using (5.5), (5.4) can be rewritten as follow

$$\rho = \frac{1}{T} \int_0^T \text{tr} \{P(t)\} dt . \quad (5.6)$$

Then, according to (5.2), $\text{tr} \{P(t)\} = \lambda(f(x, y) + y \frac{\partial f}{\partial y})$ and thus the characteristic exponent can be calculated as follows

$$\rho = \frac{\lambda}{T} \int_0^T f(x(t), \dot{x}(t)) + \dot{x}(t) f_x(x(t), \dot{x}(t)) dt . \quad (5.7)$$

It follows from the Floquet theory that the limit cycle is stable if the characteristic exponent $\rho < 0$ and is unstable if $\rho > 0$.

6. Result

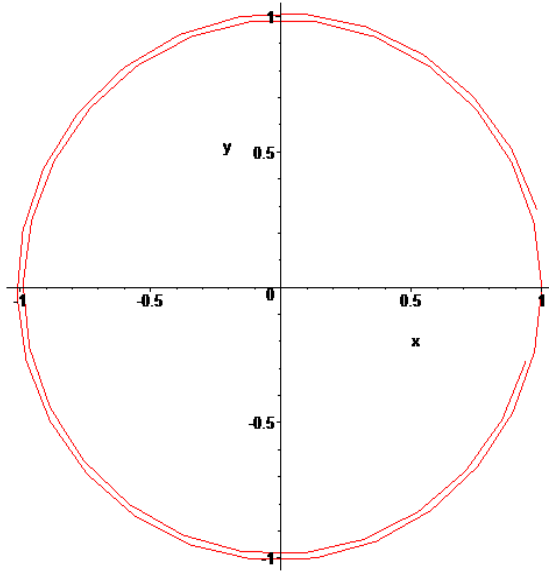
6.1 Results for the method of multiple scale

From the Maple program, we calculated the second approximation of u and hence the

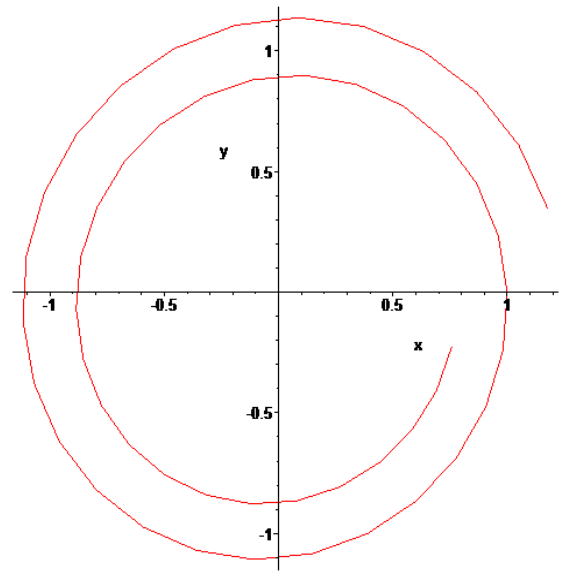
solution of equation (4.1.5) is given as follows

$$\begin{aligned}
 u(t) = & \frac{2 \cos\left(-\frac{1}{16} \varepsilon^2 t + t\right)}{\sqrt{1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}}} \\
 & + \left[-\frac{7}{8} \frac{1}{\left(1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}\right)^{\left(\frac{3}{2}\right)}} + \frac{1}{4} \frac{\ln\left(\frac{2}{\sqrt{1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}}}\right)}{\sqrt{1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}}} - \frac{2\left(-\frac{13}{64} a_0^2 + \frac{1}{8} \ln(a_0)\right)}{\sqrt{1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}}} \right] \sin\left(-\frac{1}{16} \varepsilon^2 t + t\right) \varepsilon \\
 & - \frac{1}{32} \frac{\sin\left(-\frac{3}{16} \varepsilon^2 t + 3t\right)}{\left(1 + \left(-1 + \frac{4}{a_0^2}\right) e^{(-\varepsilon t)}\right)^{\left(\frac{3}{2}\right)}} \varepsilon + O(\varepsilon^2). \tag{6.1.1}
 \end{aligned}$$

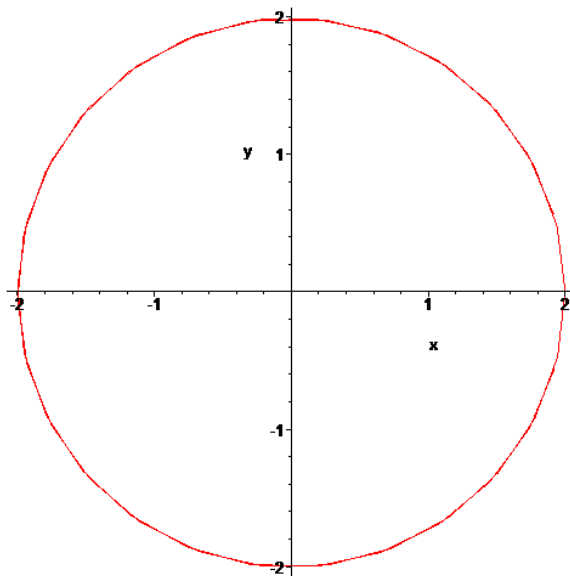
By comparing the results obtained from the method of multiple scales (using Maple) and numerical integration (using Matlab), we see that those using the method of multiple scales (Fig. 1(a-d)) are nearly the same as those using numerical integration (Fig. 1(e-h)) when the parameter ε is sufficiently small ($0 < \varepsilon \ll 1$). However, when $\varepsilon \geq 1$, the former method (Fig. 1(i-j)) gives results that are quite different from those of the latter method (Fig. 1(k-l)).



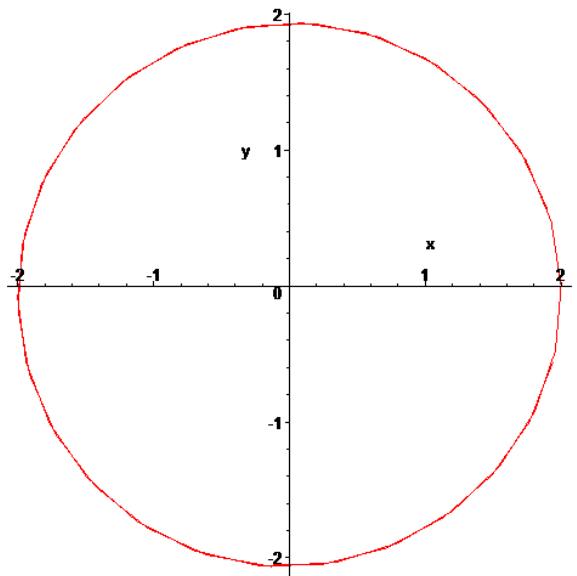
1(a) ($\varepsilon = 0.01, a_0 = 1$)



1(b) ($\varepsilon = 0.1, a_0 = 1$)

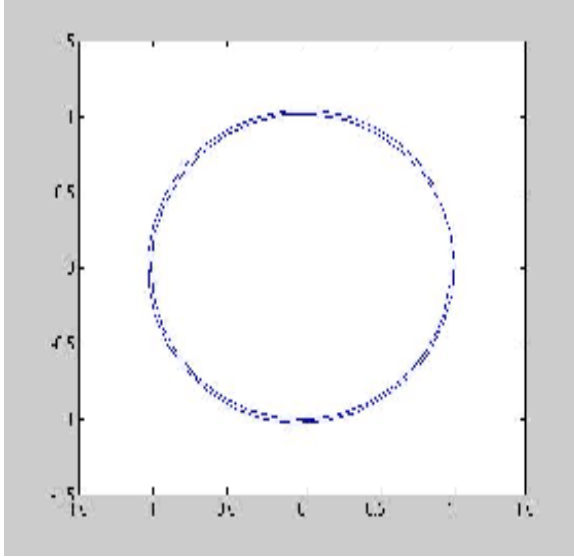


1(c) ($\varepsilon = 0.01, a_0 = 2$)

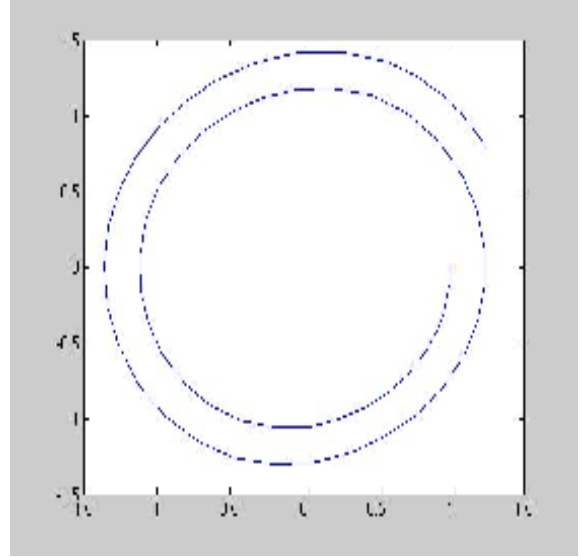


1(d) ($\varepsilon = 0.1, a_0 = 2$)

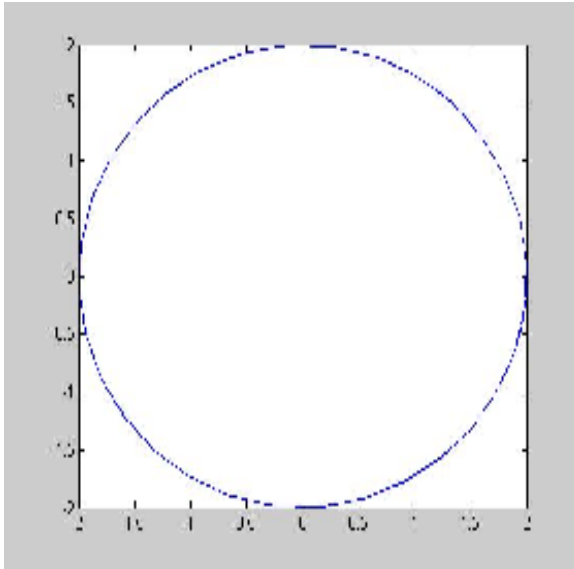
(Fig. 1(a)-Fig. 1(d)) Limit cycle of (4.1.5): the method of multiple scales



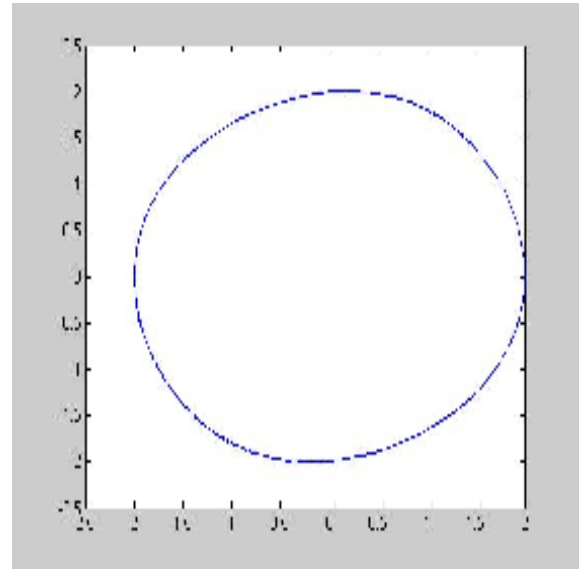
1(e) ($\varepsilon = 0.01, a_0 = 1$)



1(f) ($\varepsilon = 0.1, a_0 = 1$)

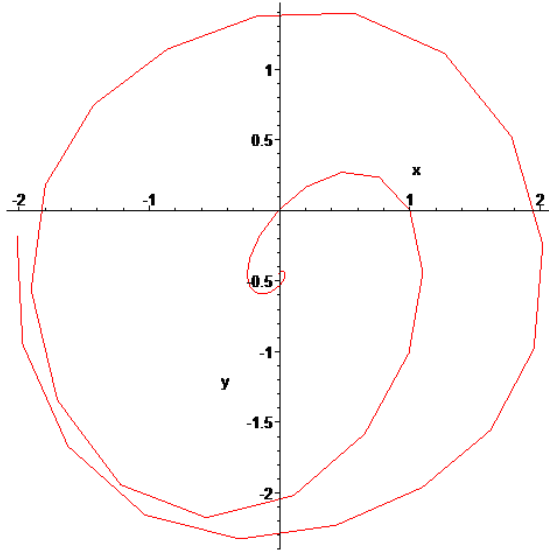


1(g) ($\varepsilon = 0.01, a_0 = 2$)

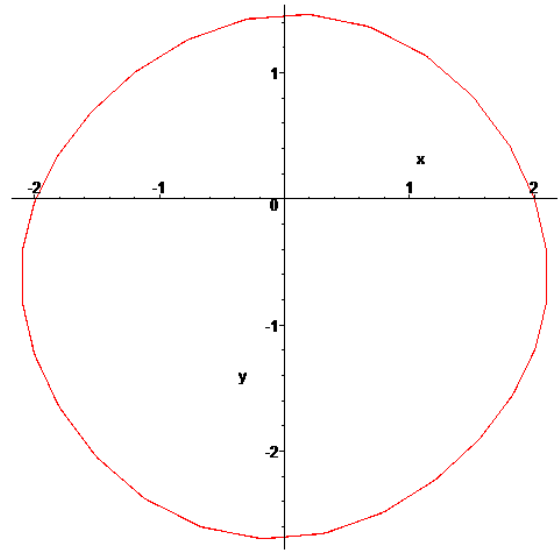


1(h) ($\varepsilon = 0.1, a_0 = 2$)

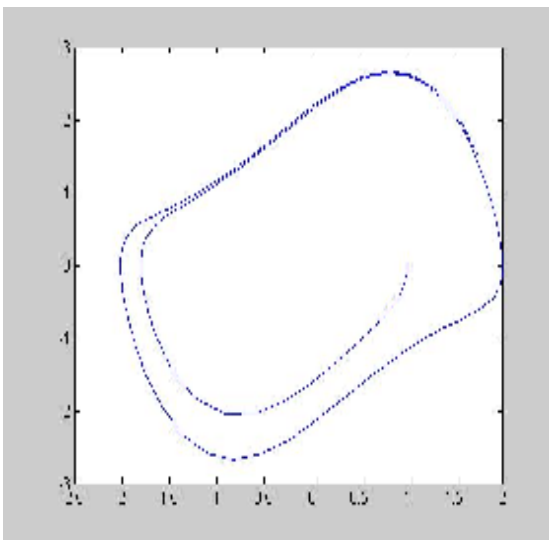
(Fig. 1(e)-Fig. 1(h)) Limit cycle of (4.1.5): Runge-Kutta method



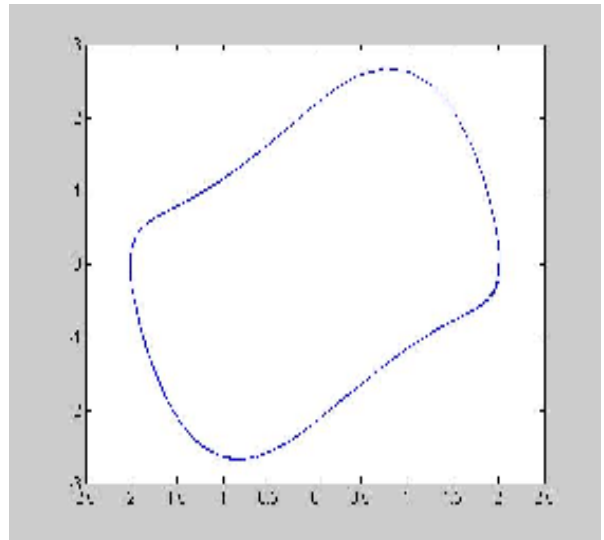
1(i) ($\varepsilon = 1, a_0 = 1$)



1(j) ($\varepsilon = 1, a_0 = 2$)



1(k) ($\varepsilon = 1, a_0 = 1$)



1(l) ($\varepsilon = 1, a_0 = 2$)

(Fig. 1(i)-Fig. 1(j)) Limit cycle of (4.1.5): the method of multiple scales

(Fig. 1(k)-Fig. 1(l)) Limit cycle of (4.1.5): Runge-Kutta method

Thus, we find that the method of multiple scales (or the classical perturbation method) is not suitable for solving limit cycle problems when the parameter ε is not small (or when $\varepsilon \geq 1$).

6.2 Results for the perturbation-incremental method

6.2.a. The generalized Van der Pol oscillator

We first consider the generalized van der Pol oscillator of the form

$$\ddot{x} + x + x^2 = \lambda(\mu + x - x^2)\dot{x}, \quad \mu \text{ is any arbitrary constant.} \quad (6.2.a.1)$$

In particular, we substitute $f(x, \dot{x}) = \mu + x - x^2$ and $g(x) = x + x^2$ into (4.2.1). Hence,

from (4.2.8), we obtain $v(x) = \frac{1}{2}x^2 + \frac{1}{3}x^3$. Also, from equations (4.2.6), (4.2.7), (4.2.12)

and (4.2.13), we obtain

$$\tilde{v}(a, b, \varphi) = -(b + \frac{1}{2})\sin^2 \varphi + \frac{b}{a}(b+1)(\cos \varphi - 1) + \frac{1}{3}a(\cos^3 \varphi - 1), \quad (6.2.a.2)$$

$$\tilde{f}(a, b, \Phi, \varphi) = \left[\mu + a \cos \varphi + b - (a \cos \varphi + b)^2 \right] \Phi \sin^2 \varphi, \quad (6.2.a.3)$$

$$\Phi_0(\varphi) = \left[1 + \frac{2}{3}a_0 \cos \varphi + 2b_0 \right]^{\frac{1}{2}}, \quad (6.2.a.4)$$

$$b_0 = \frac{1}{2} \left(\sqrt{1 - \frac{4}{3}a_0^2} - 1 \right). \quad (6.2.a.5)$$

For the first step (the perturbation method), we use (4.2.14) together with (6.2.a.4) and (6.2.a.5) to obtain the initial solution. In, particular, we choose $\mu = 0.15$. By substituting (6.2.a.4) and (6.2.a.5) into (4.2.14) and applying the Simpson's Rule, we get

$$a_0 = 0.532, \quad b_0 = -0.1055. \quad (6.2.a.6)$$

Thus,

$$\Phi_0 = [0.789 + 0.3547 \cos \varphi]^{\frac{1}{2}}. \quad (6.2.a.7)$$

Once we get the initial solution, we can apply it to the parameter incremental method. Here, we choose $M = 10$, $\lambda_0 = 0$ and $\Delta\lambda = 0.5$. After 20 successive increments of $\Delta\lambda = 0.5$ starting from $\lambda_0 = 0$, the results for $\lambda = 0.5, 10$ are given, respectively, in Tables 1 and 2. Also, a comparison of the results obtained by the perturbation-incremental method and the numerical integration using the Runge-Kutta method is shown in Fig.2 and 3. We see that the results are identical when $\lambda = 10$.

Table 1. Amplitude and bias of the limit cycle and Fourier coefficients of $\Phi(\varphi)$: $\lambda = 0.5$, $\alpha = 0.5352$, $b = -0.10514$

j	p _i	q _i
0	8.7856e-001	0.0000e+000
1	2.0583e-001	1.0451e-001
2	-1.4272e-002	-2.0498e-002
3	2.1238e-003	1.0365e-003
4	-3.0790e-004	-2.5417e-004
5	4.0804e-005	6.1918e-005
6	-4.6573e-006	-1.4434e-005
7	1.0289e-007	3.2502e-006
8	1.7769e-007	-7.0594e-007
9	-9.7644e-008	1.4196e-007
10	3.2182e-008	-2.5137e-008

Table2. Amplitude and bias of the limit cycle and
Fourier coefficients of $\Phi(\varphi)$: $\lambda=10$, $\alpha=1.2512$, $b=0.36861$

j	p_i	q_i
0	1.9188e+000	0.0000e+000
1	2.4668e-001	8.7727e-001
2	-3.8075e-001	-1.7007e+000
3	2.8615e-001	-1.2823e-001
4	-2.7121e-001	3.0623e-002
5	8.2138e-002	2.1290e-002
6	-4.9097e-002	-2.6750e-002
7	2.3713e-002	-7.2107e-004
8	-1.8923e-002	-4.5435e-003
9	1.2251e-002	3.2906e-003
10	-5.1821e-003	-6.0458e-003

Fig. 2. Limit cycle of equation (6.2.a.1) with $\mu=0.15$, for $\lambda=0.5$.
—, Runge-Kutta method; $\times$, Perturbation-Incremental method.

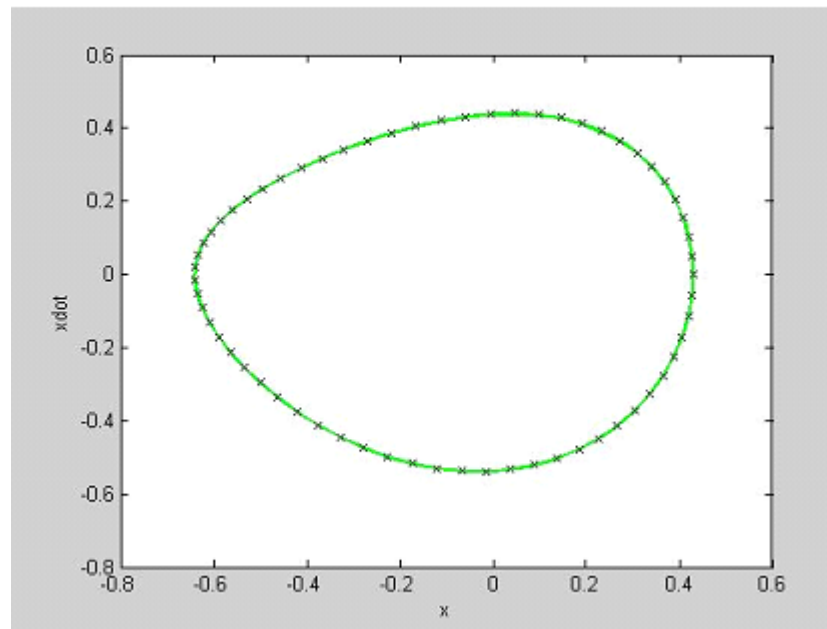
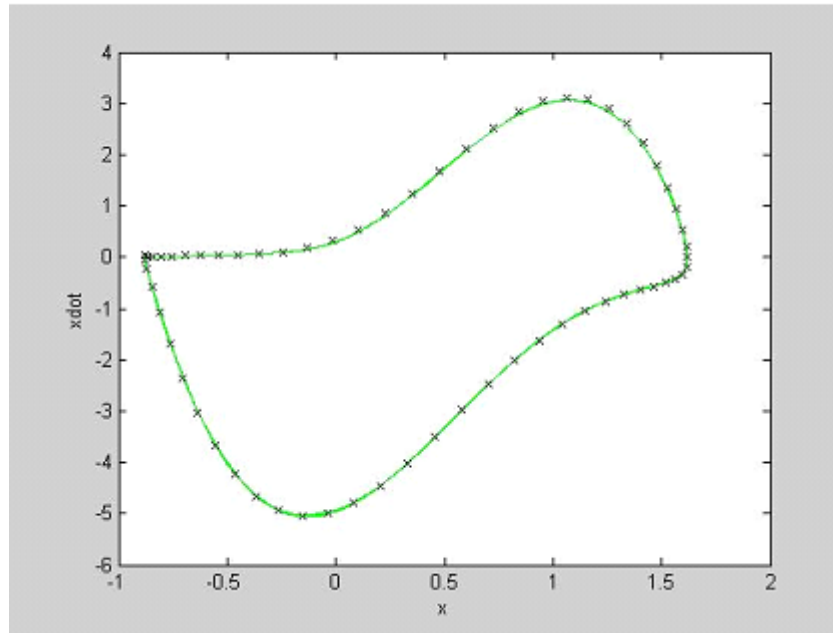


Fig. 3. Limit cycle of equation (6.2.a.1) with $\mu = 0.15$, for $\lambda = 10$.
 —, Runge-Kutta method; $\times$, Perturbation-Incremental method.

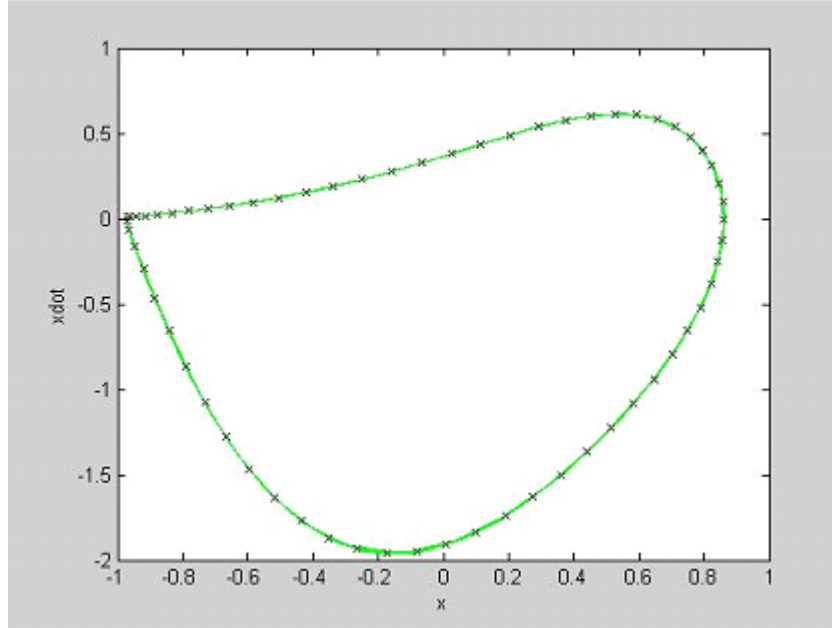


Next, we consider the case when $\mu = 0.2$. We cannot find any limit cycle when $\lambda \geq 4$. In particular, when $\mu \geq 0.3$, we cannot find any limit cycle for any value of λ . When the parameter μ gets larger, the limit cycle will also get larger and finally it will be large enough to hit a fixed point and become a homoclinic orbit (homoclinic orbit will not be discussed in this report) (see Fig.4 in which (0,-1) is a fixed point). After this, the limit cycle will disappear for any value of μ ($\mu \geq 0.3$).

Table3. Amplitude and bias of the limit cycle and
 Fourier coefficients of $\Phi(\varphi)$: $\lambda = 3$, $a = 0.91712$, $b = -0.055855$

j	p_i	q_i
0	1.0713e+000	0.0000e+000
1	3.7421e-001	8.6442e-001
2	-2.0969e-001	-2.5330e-001
3	1.2041e-001	-8.8596e-003
4	-5.0228e-002	-8.2177e-003
5	2.3244e-002	6.8676e-003
6	-1.3033e-002	-4.7860e-003
7	7.5573e-003	3.7200e-003
8	-4.0265e-003	-3.5113e-003
9	1.6731e-003	2.3433e-003
10	-5.4350e-005	-1.5274e-003

Fig. 4. Limit cycle of equation (6.2.a.1) with $\mu=0.2$, for $\lambda=3$.
 —, Runge-Kutta method; $\times$, Perturbation-Incremental method.



6.2.b. The generalized Rayleigh oscillator

We next consider the generalized Rayleigh oscillator of the form

$$\ddot{x} + x^3 = \lambda(\mu - \dot{x}^2)\dot{x}, \quad \mu \text{ is any arbitrary constant.} \quad (6.2.b.1)$$

For this example, we let $f(x, \dot{x}) = \mu - \dot{x}^2$, $g(x) = x^3$ and hence $v(x) = \frac{1}{4}x^4$. The limit cycle is symmetric about the origin. Thus, $b = 0$ and we rewrite (4.2.20) as follows

$$\Phi(\varphi) = \sum_{j=0}^M (p_{2j} \cos 2j\varphi + q_{2j} \sin 2j\varphi). \quad (6.2.b.2)$$

First, we seek the condition for (6.2.b.1) to have positive amplitude a_0 . By using (4.2.14),

μ can be related to amplitude a_0 as

$$\mu = \frac{a_0^2 \int_0^{2\pi} \Phi_0^3(\varphi) \sin^4 \varphi d\varphi}{\int_0^{2\pi} \Phi_0(\varphi) \sin^2 \varphi d\varphi}. \quad (6.2.b.3)$$

Fig. 5 shows the relationship between parameter μ and amplitude a_0 .

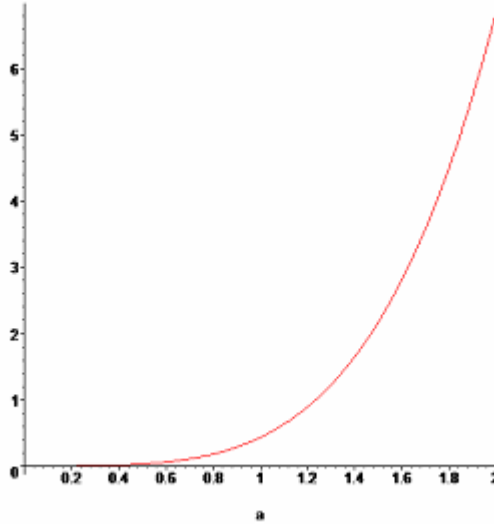


Fig. 5. The parameter μ vs amplitude a_0 of the limit cycle of equation (6.2.b.1)

In particular, we choose $\mu = 2.5$. By using (6.2.b.3) and (4.2.12), we obtain the amplitude and the solution, respectively, as

$$a_0 = 1.5541, \quad (6.2.b.4)$$

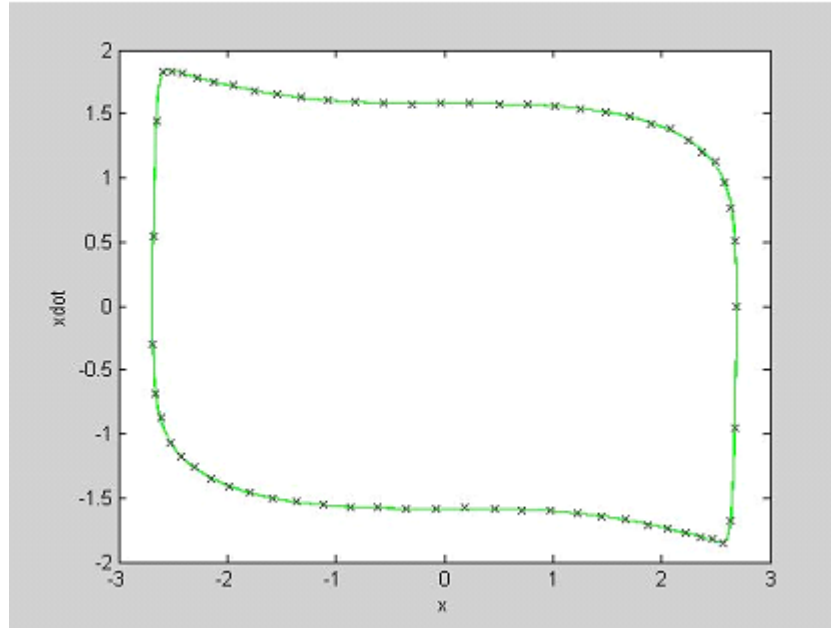
$$\Phi_0(\varphi) = a_0 \left[\frac{1}{2} (1 + \cos^2 \theta) \right]^{\frac{1}{2}}. \quad (6.2.b.5)$$

Then, the procedures are actually very similar to the pervious example. Therefore, we give the result directly as in Fig. 6 and Table 4.

Table4. Amplitude and bias of the limit cycle and
Fourier coefficients of $\Phi(\varphi)$: $\lambda=10$, $\alpha=2.6844$, $b=0$

j	p_{2j}	q_{2j}
0	1.1765e+000	0.0000e+000
1	8.8384e-001	2.0266e-001
2	4.6272e-001	2.5168e-001
3	2.6067e-001	2.1670e-001
4	1.4567e-001	1.7481e-001
5	7.7775e-002	1.3312e-001
6	3.8958e-002	9.5945e-002
7	1.8240e-002	6.4921e-002
8	8.3542e-003	4.0256e-002
9	4.1888e-003	2.1508e-002
10	2.0760e-003	8.2137e-003

Fig. 6. Limit cycle of equation (6.2.b.1) with $\mu=2.5$, for $\lambda=10$.
—, Runge-Kutta method; $\times$, Perturbation-Incremental method.



An interesting analysis for the generalized Rayleigh oscillator is that the limit cycle is always stable for any arbitrary value of parameter $\mu > 0$. To explain this, we analyze the stability of the limit cycle. Recall from section 5 that we can calculate the characteristic

exponent of the limit cycle by equation (5.7). Thus, we obtain

$$\rho = \frac{\lambda}{T} \int_0^T (\mu - \dot{x}^2(t)) - 2\dot{x}^2(t) dt. \quad (6.2.b.6)$$

By using $x = a \cos \varphi + b$, we can rewrite (6.2.b.6) as

$$\rho = \frac{\lambda}{T} \int_0^{2\pi} \frac{\mu - 3a_0^2 \Phi_0^2(\varphi) \sin^2 \varphi}{\Phi_0(\varphi)} d\varphi = \frac{2\lambda}{T} \rho_0(a_0), \quad (6.2.b.7)$$

with

$$\rho_0(a_0) = \int_0^\pi \frac{\mu - 3a_0^2 \Phi_0^2(\varphi) \sin^2 \varphi}{\Phi_0(\varphi)} d\varphi. \quad (6.2.b.8)$$

From Fig. 7, we find that the parameter ρ_0 is always less than zero and so is the characteristic exponent ρ . Thus, the limit cycle of (6.2.b.1) is always stable for $\mu > 0$.

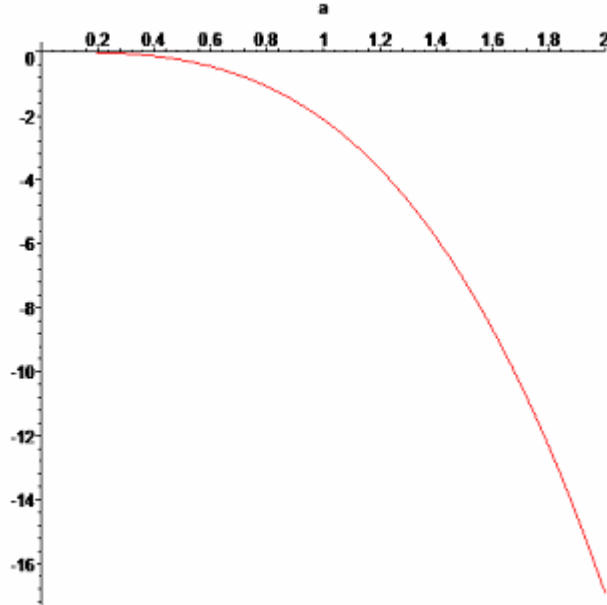


Fig. 7. The parameter ρ_0 vs amplitude a_0 of the limit cycle of equation (6.2.b.1)

6.2.c. The generalized Liénard oscillator

Finally, we consider the generalized Liénard oscillator of the form

$$\ddot{x} + x^3 = \lambda(\mu + x^2 - x^4)\dot{x}, \quad \mu \text{ is any arbitrary constant.} \quad (6.2.c.1)$$

For this example, we let $f(x, \dot{x}) = \mu + x^2 - x^4$, $g(x) = x^3$ and hence $v(x) = \frac{1}{4}x^4$. The limit

cycle is symmetric about the origin. Thus, $b = 0$ and we rewrite (4.2.20) as

$$\Phi(\varphi) = \sum_{j=0}^M (p_{2j} \cos 2j\varphi + q_{2j} \sin 2j\varphi). \quad (6.2.c.2)$$

First, we seek the condition for (6.2.b.1) to have a positive amplitude a_0 . By using

(4.2.14), μ can be related to amplitude a_0 as

$$\mu = \frac{a_0^4 \int_0^{2\pi} \Phi_0(\varphi) \cos^4 \varphi \sin^2 \varphi d\varphi - a_0^2 \int_0^{2\pi} \Phi_0(\varphi) \cos^2 \varphi \sin^2 \varphi d\varphi}{\int_0^{2\pi} \Phi_0(\varphi) \sin^2 \varphi d\varphi}. \quad (6.2.c.3)$$

Fig. 8 shows the relationship between μ and a_0 .

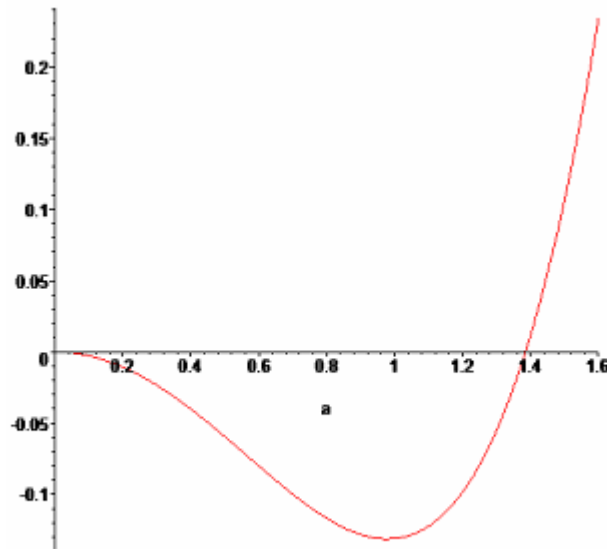


Fig. 8. The parameter μ vs amplitude a_0 of the limit cycle of equation (6.2.c.1)

For the case $\mu = -0.13$, we obtain from Fig. 8 two different positive amplitudes $a_0^{(1)}$ and $a_0^{(2)}$ which correspond to two solutions $\Phi_0^{(1)}(\varphi)$ and $\Phi_0^{(2)}(\varphi)$:

$$a_0^{(1)} = 0.9249986552, \quad \Phi_0(\varphi) = a_0^{(1)} \left[\frac{1}{2} (1 + \cos^2 \theta) \right]^{\frac{1}{2}}, \quad (6.2.c.4)$$

$$a_0^{(2)} = 1.031287122, \quad \Phi_0(\varphi) = a_0^{(2)} \left[\frac{1}{2} (1 + \cos^2 \theta) \right]^{\frac{1}{2}}. \quad (6.2.c.5)$$

In particular, (6.2.c.4) is an unstable limit cycle while (6.2.c.5) is a stable limit cycle. Fig. 9 shows the relationship between the characteristic exponent ρ and amplitude a_0 .

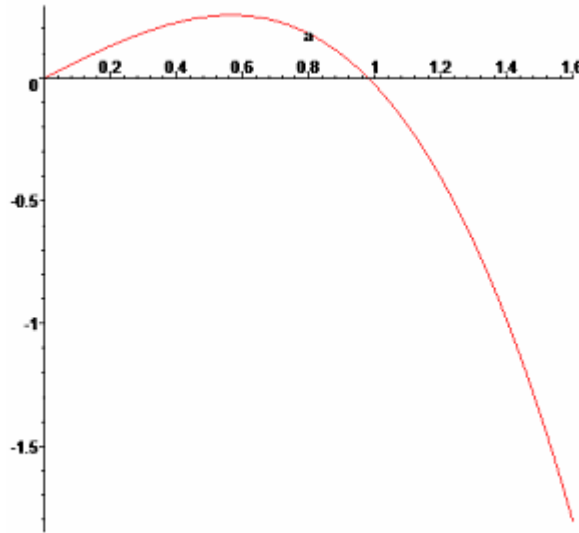


Fig. 9. The parameter ρ_0 vs amplitude a_0 of the limit cycle of equation (6.2.c.1)

From Fig. 9, we find that $\rho > 0$ (the limit cycle is unstable) when $a_0 < a_0^*$ while $\rho < 0$ (the limit cycle is stable) when $a_0 > a_0^*$ where $a_0^* = 0.9796$. That is the reason why (6.2.c.4) is an unstable limit cycle while (6.2.c.5) is stable.

Fig. 10 and Table 5 show the result for the case of unstable limit cycle of (6.2.c.1) with $\mu = -0.13$.

Table 5. Amplitude and bias of the limit cycle and Fourier coefficients of $\Phi^{(1)}(\varphi)$: $\lambda = 10$, $\alpha = 0.94729$, $b = 0$

j	p_{2j}	q_{2j}
0	8.2661e-001	0.0000e+000
1	1.2247e-001	2.7884e-001
2	-4.0795e-003	-1.7596e-001
3	3.6070e-003	4.1418e-003
4	-1.1952e-003	-1.0779e-003
5	-2.9265e-004	4.9148e-004
6	1.6558e-004	-9.2489e-005
7	-7.1866e-005	-3.2700e-005
8	6.2148e-006	2.4998e-005
9	1.2185e-006	-1.0034e-005
10	-5.7799e-006	7.9721e-007

Fig. 10. **Unstable** limit cycle of equation (6.2.c.1) with $\mu = -0.13$, for $\lambda = 10$.
—, Runge-Kutta method; ×, Perturbation-Incremental method.

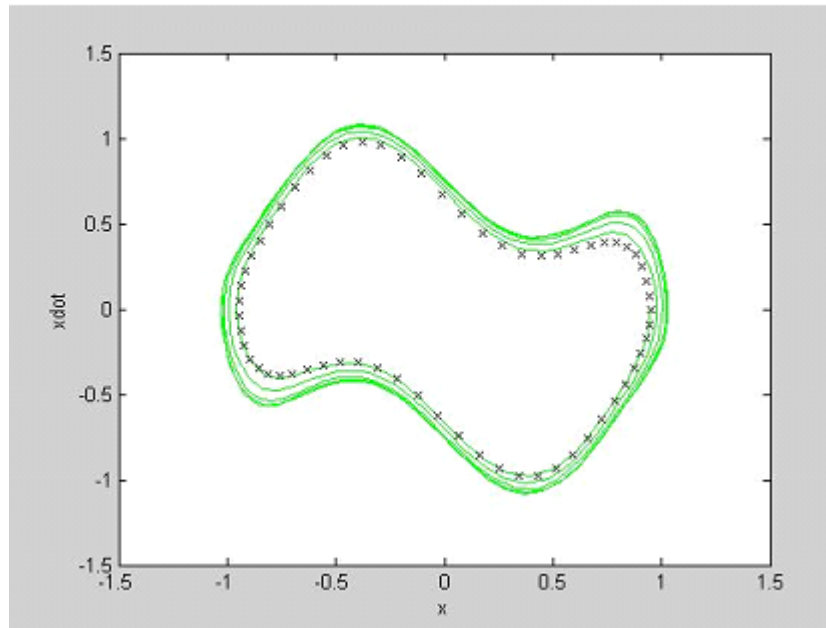
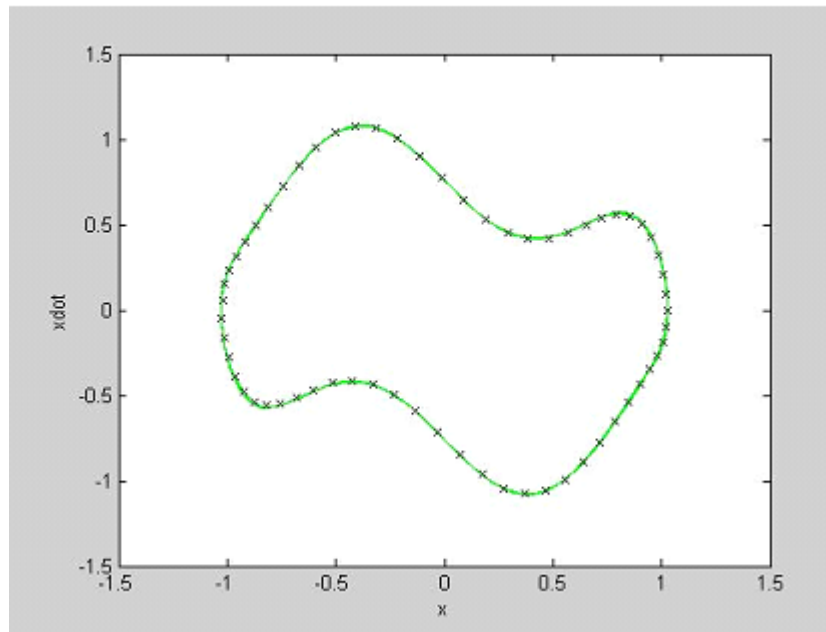


Fig. 11 and Table 6 show the result for the case of stable limit cycle of (6.2.c.1) with $\mu = -0.13$.

Table6. Amplitude and bias of the limit cycle and Fourier coefficients of $\Phi^{(2)}(\varphi) : \lambda = 10, \alpha = 1.024, \delta = 0$		
j	p_{2j}	q_{2j}
0	8.8821e-001	0.0000e+000
1	1.4325e-001	1.5663e-001
2	-5.0542e-003	-2.3585e-001
3	-3.2236e-005	5.4146e-003
4	-2.2315e-003	-7.8375e-004
5	-2.5410e-004	2.9648e-004
6	1.1062e-004	-1.8604e-004
7	-6.2958e-005	-7.0164e-006
8	1.8705e-005	1.6676e-005
9	-3.8733e-006	-9.2941e-006
10	-4.2953e-006	3.4392e-006

Fig. 11. **Stable** limit cycle of equation (6.2.c.1) with $\mu = -0.13$, for $\lambda = 10$.
—, Runge-Kutta method; ×, Perturbation-Incremental method.



An interesting result is that when the parameter μ gets larger, the separation between the unstable limit cycle and the stable one increases or equivalently the unstable limit cycle approach as the origin while the stable one moves far away from the origin. In particular,

when $\mu \geq 0$, the unstable limit cycle disappears leaving one and only one limit cycle which is stable. Fig. 12-16 show the limit cycles for different values of μ to illustrate the situation.

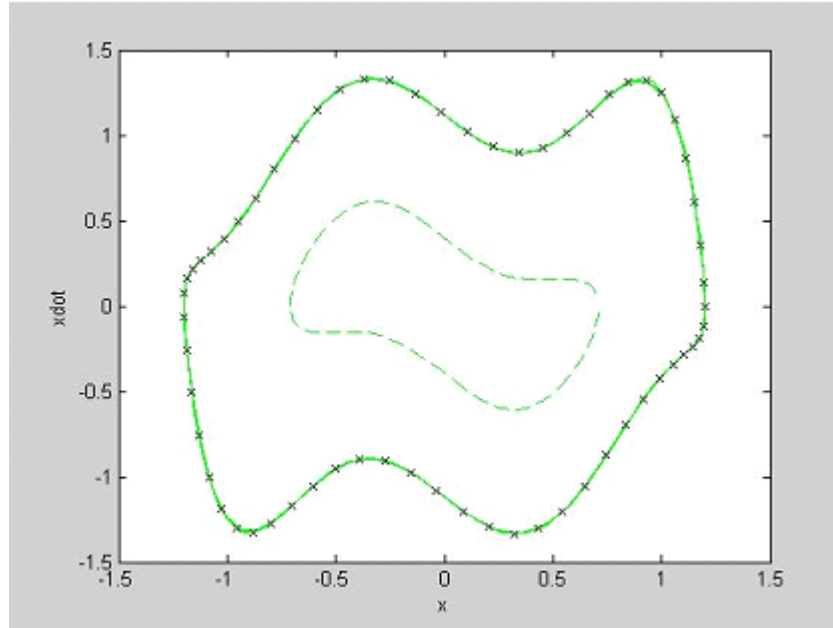


Fig. 12. $\mu = -0.1$, for $\lambda = 10$.

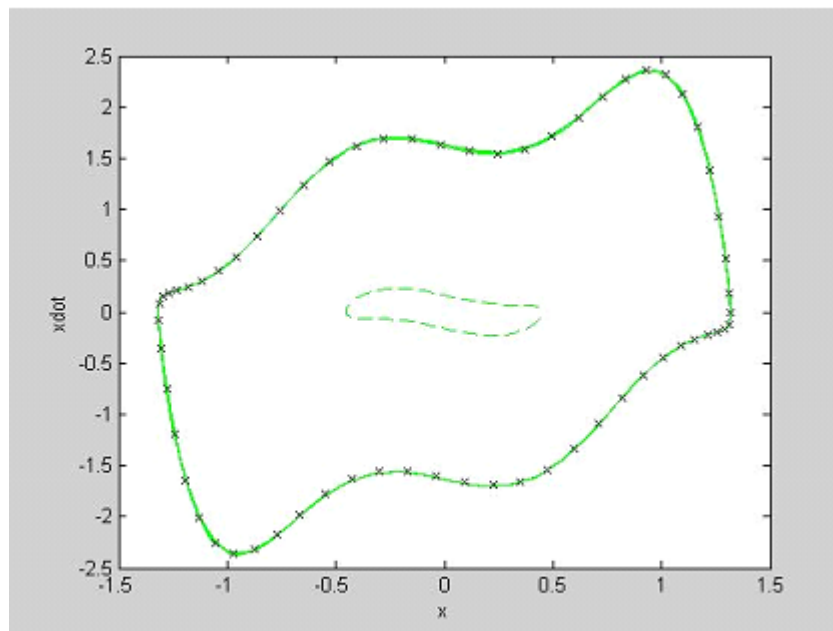


Fig. 13. $\mu = -0.05$, for $\lambda = 10$.

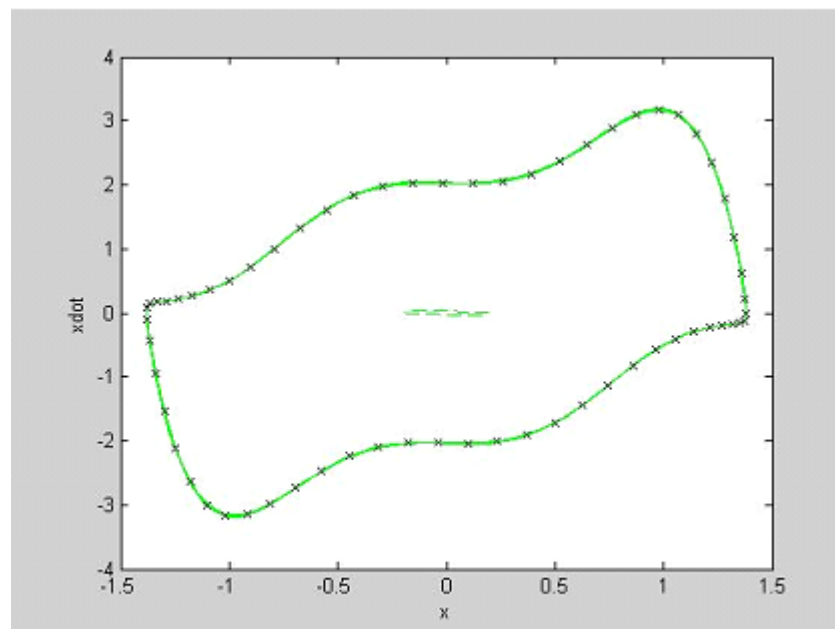


Fig. 14. $\mu = -0.01$, for $\lambda = 10$.

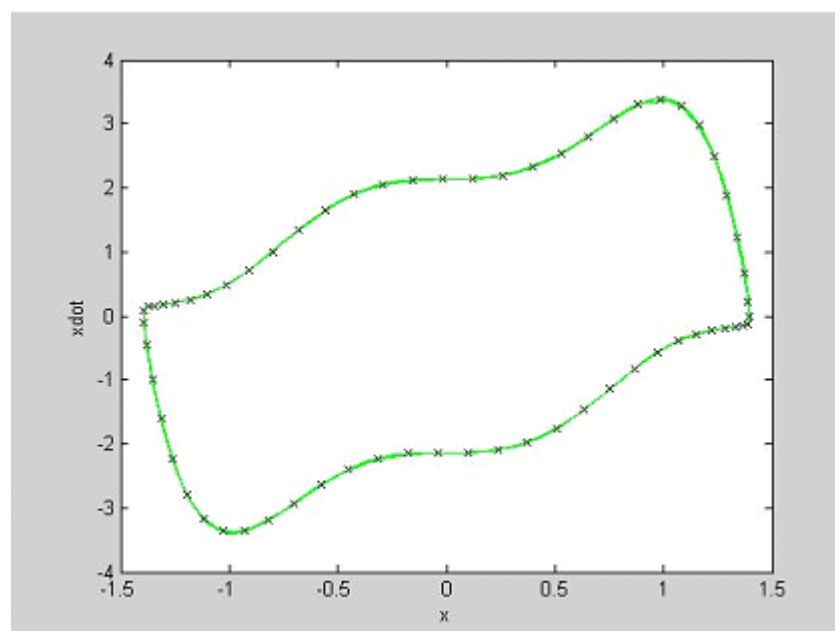


Fig. 15. $\mu = 0$, for $\lambda = 10$.

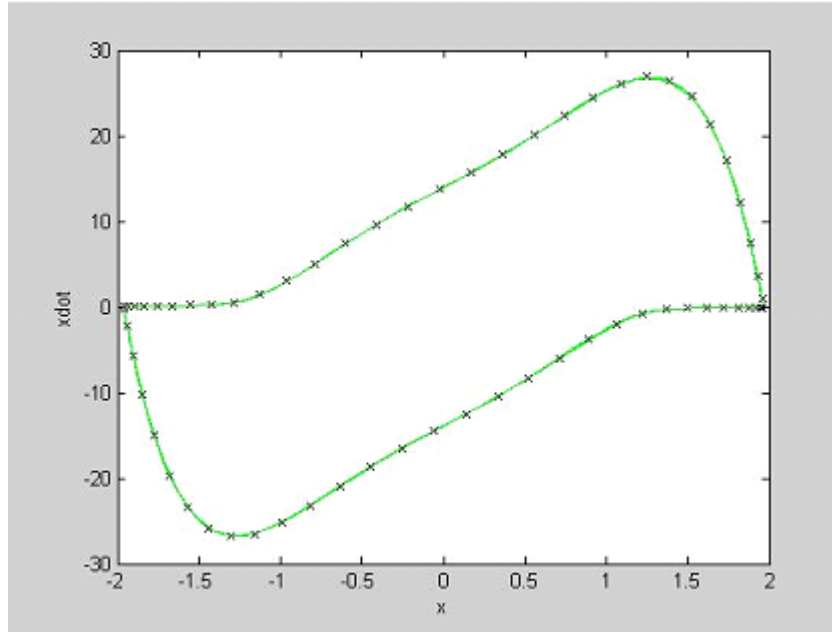


Fig. 16. $\mu=1$, for $\lambda=10$.

(Fig. 12-Fig. 16) Limit cycle of (6.2.c.1): —, stable limit cycle, Runge-Kutta method;
 ×, stable limit cycle, Perturbation-Incremental (PI) method;
 --- unstable limit cycle, PI method.

We can explain the previous results (Fig. 12 – Fig. 16) using Fig. 8. When the parameter μ is below the minimum of the curve (here, we denote the minimum of the curve by μ^* where $\mu^* = -0.1315$), we find that there is no real root and, consequently, (6.2.c.1) does not have any limit cycle. However, for $\mu^* < \mu < 0$, we see that there are two roots and, consequently, (6.2.c.1) has exactly two limit cycles. We then use Fig. 9 to determine their stabilities. Finally, when $\mu \geq 0$, we find that there is exactly one root and hence (6.2.c.1) has one and only one limit cycle which is stable from Fig. 9.

7. Discussion

In section 6, we find that the method of multiple scales (or the classical perturbation method) is not suitable for solving the limit cycle problems when the parameter ε is not small. Recall in section 4, the solution is assumed to be of the form (4.1.8), which is a power series of the parameter ε . In particular, it has the following form

$$u(t; \varepsilon) = u(T_0, T_1, T_2; \varepsilon) = u_0(T_0, T_1, T_2) + \varepsilon u_1(T_0, T_1, T_2) + \varepsilon^2 u_2(T_0, T_1, T_2) + O(\varepsilon^3). \quad (7.1)$$

Thus, we see from (7.1) that the solution is only approximated by a finite number of terms which is in the power of ε with some error terms in $O(\varepsilon^3)$. Hence, when $\varepsilon \geq 1$, the error terms in (7.1) will grow exponentially fast. This explains why the method of multiple scales is only useful when parameter ε is sufficiently small ($0 < \varepsilon \ll 1$).

Another disadvantage of this method is that it cannot solve strongly non-linear oscillators for a limit cycle. Even though we apply the method of multiple scales to (6.2.1), the purpose of multiple scales still cannot be achieved. For example, we consider the collection of $O(\varepsilon^0)$ on the both sides of (6.2.1) which have already applied the multiple scales i.e.

$$\frac{\partial^2 x_0}{\partial T_0^2} + x_0 + x_0^2 = 0 \quad \text{with} \quad x_0 = x_0(T_0, T_1, T_2). \quad (7.2)$$

Equation (7.2) is still a non-linear ordinary differential equation which we cannot solve analytically. In general, the non-linear function $g(x)$ in (4.2.1) prevents us from using the

method of multiple scales since this non-linear function generates non-linear terms in the ordinary differential equation such as (7.2) after the method is applied. This explains why the method of multiple scales is not useful in solving the strongly non-linear oscillators of the form (4.2.1).

From section 6, for the particular problem of the generalized Liénard oscillator, we find that the numerical methods such as the Runge-Kutta method (the one we are using in this report) are unable to obtain unstable limit cycle. Since the Runge-Kutta (RK) method is a one-step explicit method which means the new solution at the time t_{j+1} depends only on the previous solution at time t_j . Thus, if the limit cycle is unstable, every solution obtained by the RK method will leave the path of the limit cycle and consequently we cannot obtain the unstable limit cycle.

Finally, we find that the Perturbation-Incremental method can overcome the difficulties of the method of multiple scales and the numerical method (those we have already discussed in this section).

8. Conclusion

The method of multiple scales is applied to solve the weakly non-linear oscillators for the limit cycle problem. However, the classical perturbation method such as the one we have presented in this report (the method of multiple scales) is not accurate when the parameter ε is not sufficiently small (usually we require $0 < \varepsilon \ll 1$). We also indicate that the method of multiple scales is not suitable to solve the strongly non-linear oscillators even though we apply the method to such problems. Thus, we introduce the perturbation-incremental method to the study of strongly non-linear oscillators for the limit cycle problem. The perturbation-incremental is a combination of the analytical and the numerical methods. For the analytical aspect, we use the perturbation step to obtain the initial solution of a limit cycle. This overcomes the difficulty in the numerical integration method that it usually requires a guess of initial condition. For the numerical aspect, we use the incremental step to obtain the solution of a limit cycle for arbitrary parameter $\lambda > 0$. This also overcomes the difficulty in the classical perturbation method that it requires the parameter to be sufficiently small. The stability of a limit cycle can be determined using the Floquet theory by calculating the characteristic exponent directly from the solution.

9. References

1. Alan, W. Bush, *Perturbation Method for Engineers and Scientists*, Teesside Polytechnic, U.K., 1990.
2. Nayfeh, A.H., *Perturbation Methods*, John Wiley & Sons, Inc., New York, 2000.
3. Martha, L. Abell and James, P. Braselton, *Maple V[®] By Example*, Academic Press, 1999.
4. Chan, H. S. Y., Chung, K. W. and Xu, Z., *A Perturbation-Incremental Method for Strongly Non-Linear Oscillators*, *International Journal of Non-Linear Mechanics* **31**, 1996, 59-72.
5. Steven, C. Chapra, *Applied Numerical Methods with Matlab for Engineers and Scientists*, McGraw. Hill, 2005.
6. Jordan, D. W. and Smith, P., *Nonlinear Ordinary Differential Equations*, Oxford University Press, New York, 1987.
7. Paul, Glendinning, *Stability, Instability and Chaos: an introduction to the theory of nonlinear differential equations*, Cambridge University Press, 1994.

10. Appendix

$$A_0 = \alpha_0 - \lambda \sum_{k=1}^M \frac{1}{k} \delta_{2,k} ,$$

$$A_i = \alpha_i + \frac{\lambda \delta_{2,i}}{i} ,$$

$$A_{M+i} = -\frac{\lambda \gamma_{2,i}}{i} ,$$

$$A_{2M+1} = \sum_{k=0}^M (-1)^k \alpha_k - \lambda \sum_{k=1}^M \frac{1}{k} [1 - (-1)^k] \delta_{2,k} ,$$

$$A_{2M+2} = \gamma_{2,0} ,$$

$$B_0 = \beta_0 - \lambda \sum_{k=1}^M \frac{1}{k} \delta_{3,k} ,$$

$$B_i = \beta_i + \frac{\lambda \delta_{3,i}}{i} ,$$

$$B_{M+i} = -\frac{\lambda \gamma_{3,i}}{i} ,$$

$$B_{2M+1} = \sum_{k=0}^M (-1)^k \beta_k - \lambda \sum_{k=1}^M \frac{1}{k} [1 - (-1)^k] \delta_{3,k} ,$$

$$B_{2M+2} = \gamma_{3,0} ,$$

$$A_{0,j} = \frac{1}{2} (\zeta_{1,-j} + \zeta_{1,j}) - \frac{1}{2} \lambda \sum_{k=1}^M \frac{1}{k} (\delta_{4,k-j} - \delta_{4,j-k} + \delta_{4,j+k}) ,$$

$$A_{i,j} = \frac{1}{2} (\zeta_{1,i-j} + \zeta_{1,j-i} + \zeta_{1,j+i}) + \frac{\lambda}{2i} (\delta_{4,i-j} - \delta_{4,j-i} + \delta_{4,j+i}) ,$$

$$A_{M+i,j} = \frac{1}{2} (\eta_{1,i-j} - \eta_{1,j-i} + \eta_{1,j+i}) - \frac{\lambda}{2i} (\gamma_{4,i-j} + \gamma_{4,j-i} + \gamma_{4,j+i}) ,$$

$$A_{2M+1,j} = -\frac{1}{2} \lambda \sum_{k=1}^M \frac{[1 - (-1)^k]}{k} (\delta_{4,k-j} - \delta_{4,j-k} + \delta_{4,j+k}) ,$$

$$A_{2M+2,j} = \frac{1}{2} (\gamma_{4,-j} + \gamma_{4,j}) ,$$

$$B_{0,j} = \frac{1}{2} (\eta_{1,j} - \eta_{1,-j}) - \frac{1}{2} \lambda \sum_{k=1}^M \frac{1}{k} (\gamma_{4,k-j} + \gamma_{4,j-k} - \gamma_{4,j+k}) ,$$

$$B_{i,j} = \frac{1}{2} (\eta_{1,j-i} + \eta_{1,j+i} - \eta_{1,i-j}) + \frac{\lambda}{2i} (\gamma_{4,i-j} + \gamma_{4,j-i} - \gamma_{4,j+i}) ,$$

$$B_{M+i,j} = \frac{1}{2}(\zeta_{1,i-j} + \zeta_{1,j-i} - \zeta_{1,j+i}) - \frac{\lambda}{2i}(\delta_{4,j-i} + \delta_{4,j+i} - \delta_{4,i-j}),$$

$$B_{2M+1,j} = -\frac{1}{2}\lambda \sum_{k=1}^M \frac{[1-(-1)^k]}{k} (\gamma_{4,k-j} + \gamma_{4,j-k} - \gamma_{4,j+k}),$$

$$B_{2M+2,j} = \frac{1}{2}(\delta_{4,j} - \delta_{4,-j}),$$

$$R_0 = -\zeta_{2,0} + \lambda \sum_{k=1}^M \frac{1}{k} \delta_{1,k},$$

$$R_i = -\zeta_{2,i} - \frac{\lambda \delta_{1,i}}{i},$$

$$R_{M+i} = -\eta_{2,i} + \frac{\lambda \gamma_{1,i}}{i},$$

$$R_{2M+1} = -\sum_{k=0}^M (-1)^k \tau_k + \lambda \sum_{k=1}^M \frac{1}{k} [1-(-1)^k] \delta_{1,k},$$

$$R_{2M+2} = -\gamma_{1,0},$$

where

$$i = 1, 2, \dots, M, j = 0, 1, 2, \dots, M,$$

$$\zeta_{1,k} = \eta_{1,k} = \gamma_{4,k} = \delta_{4,k} = 0, \text{ for } k < 0.$$

Matlab Program for the generalized Liénard oscillator

```

format long
mu=-0.1;a=1.195613137;M=20;P(3*M+6)=0;Q(3*M+3)=0;
P(1)=a;P(2)=0;P(3)=a*0.8598244315;P(4)=0;P(5)=a*.1458622585;
P(6)=0;P(7)=a*-.6276116194e-2;P(8)=0;P(9)=a*.4967276956e-3;
P(10)=0;P(11)=a*-.8957687693e-4;P(12)=0;P(13)=a*-.2027626537e-4;
P(14)=0;P(15)=a*-.2273926530e-4;P(16)=0;P(17)=a*-.1679710530e-4;
P(18)=0;P(19)=a*-.1259587300e-4;P(20)=0;P(21)=a*-.8983320421e-5;
P(22)=0;P(23)=a*-.616436998e-5;
for lambda=0.5:0.5:10
    while (1)
        for k=1:(M+1)
            if k==1
                GR(k)=-(10/32)*P(1)-P(2)+(P(2)^3)/(P(1)^2);
                BR(k)=-P(1)-(3*(P(2)^2))/P(1)-(3/2)*P(2);
            elseif k==2
                GR(k)=(3/4)*P(2)-(P(2)^3)/(P(1)^2);
                BR(k)=(3/4)*P(1)+(3*(P(2)^2))/P(1);
            elseif k==3
                GR(k)=(2/8)*P(1);BR(k)=(3/2)*P(2);
            elseif k==4
                GR(k)=(1/4)*P(2);BR(k)=(1/4)*P(1);
            elseif k==5
                GR(k)=(2/32)*P(1);BR(k)=0;
            else
                GR(k)=0;BR(k)=0;
            end
        end
    end
    for k=1:(3*M+1)
        if k==1
            GA(k)=(1/2)*mu+(1/2)*((P(2))^2)-(1/2)*((P(2))^4)...
                +(1/8)*((P(1))^2)-(3/4)*((P(1))^2)*((P(2))^2)-(1/16)*((P(1))^4);
            BM(k)=(1/4)*P(1)-(3/2)*P(1)*((P(2))^2)-(1/4)*((P(1))^3);
            BE(k)=P(2)-(3/2)*((P(1))^2)*P(2)-2*((P(2))^3);
        elseif k==2
            GA(k)=-(1/2)*((P(1))^3)*P(2)-P(1)*((P(2))^3)+(1/2)*P(1)*P(2);
            BM(k)=(1/2)*P(2)-((P(2))^3)-(3/2)*((P(1))^2)*P(2);
        end
    end
end

```

```

        BE(k)=(1/2)*P(1)-(1/2)*((P(1))^3)-3*P(1)*((P(2))^2);
elseif k==3
    GA(k)=-(1/32)*((P(1))^4)...
        -(1/2)*mu-(1/2)*((P(2))^2)+(1/2)*((P(2))^4);
    BM(k)=-(1/8)*((P(1))^3); BE(k)=-P(2)+2*((P(2))^3);
elseif k==4
    GA(k)=(1/4)*((P(1))^3)*P(2)...
        -(1/2)*P(1)*P(2)+P(1)*((P(2))^3);
    BM(k)=(3/4)*((P(1))^2)*P(2)-(1/2)*P(2)+((P(2))^3);
    BE(k)=-(1/2)*P(1)+(1/4)*((P(1))^3)+3*P(1)*((P(2))^2);
elseif k==5
    GA(k)=(1/16)*((P(1))^4)...
        -(1/8)*((P(1))^2)+(3/4)*((P(1))^2)*((P(2))^2);
    BM(k)=(1/4)*((P(1))^3)-(1/4)*P(1)+(3/2)*P(1)*((P(2))^2);
    BE(k)=(3/2)*((P(1))^2)*P(2);
elseif k==6
    GA(k)=(1/4)*((P(1))^3)*P(2);
    BM(k)=(3/4)*((P(1))^2)*P(2); BE(k)=(1/4)*((P(1))^3);
elseif k==7
    GA(k)=(1/32)*((P(1))^4);
    BM(k)=(1/8)*((P(1))^3); BE(k)=0;
else
    GA(k)=0;BM(k)=0;BE(k)=0;
end
end
for k=1:(3*M+1)
    if k==1
        u(k)=(0.5)*P(3)-0.25*P(5);v(k)=0;
    else
        if (k-1)-2>0
            u(k)=0.5*P(k-1+3)-0.25*P((k-1)-2+3)-0.25*P((k-1)+2+3);
            v(k)=0.5*Q(k-1)-0.25*Q((k-1)-2)-0.25*Q((k-1)+2);
        elseif (k-1)-2<0
            u(k)=0.5*P(k-1+3)-0.25*P((k-1)+2+3)-0.25*P(2-(k-1)+3);
            v(k)=0.5*Q(k-1)-0.25*Q((k-1)+2)+0.25*Q(2-(k-1));
        else
            u(k)=0.5*P(k-1+3)-0.25*P((k-1)+2+3)-0.5*P(2-(k-1)+3);
            v(k)=0.5*Q(k-1)-0.25*Q((k-1)+2);
        end
    end
end

```

```

end
end
end
for k=1:(2*M+1)
    if k==1
        sum1=0;sum2=0;
        for j=0:M
            if j==0
                sum1=sum1+(P(j+3)*(2*u(j+1)));
            else
                sum2=sum2+(P(j+3)*(u(j+1))+Q(j)*(v(j+1)));
            end
        end
        end
        KD(k)=0.25*(sum1+sum2);KE(k)=0;
    else
        sum1=0;sum2=0;sum3=0;sum4=0;sum5=0;sum6=0;
        for j=0:M
            if j==0
                if (k-1)-j>0
                    sum1=sum1+(P(j+3)*(u((k-1)-j+1)+u((k-1)+j+1)));
                    sum4=sum4+(P(j+3)*(v((k-1)-j+1)+v((k-1)+j+1)));
                elseif (k-1)-j<0
                    sum2=sum2+(P(j+3)*(u(j-(k-1)+1)+u((k-1)+j+1)));
                    sum5=sum5+(P(j+3)*(-v(j-(k-1)+1)+v((k-1)+j+1)));
                else
                    sum3=sum3+(P(j+3)*(2*u(j-(k-1)+1)+u((k-1)+j+1)));
                    sum6=sum6+(P(j+3)*(v((k-1)+j+1)));
                end
            end
            else
                if (k-1)-j>0
                    sum1=sum1+(P(j+3)*(u((k-1)-j+1)+u((k-1)+j+1))...
                        +Q(j)*(v((k-1)+j+1)-v((k-1)-j+1)));
                    sum4=sum4+(P(j+3)*(v((k-1)-j+1)+v((k-1)+j+1))...
                        +Q(j)*(-u((k-1)+j+1)+u((k-1)-j+1)));
                elseif (k-1)-j<0
                    sum2=sum2+(P(j+3)*(u(j-(k-1)+1)+u((k-1)+j+1))...
                        +Q(j)*(v((k-1)+j+1)+v(j-(k-1)+1)));
                    sum5=sum5+(P(j+3)*(-v(j-(k-1)+1)+v((k-1)+j+1))...

```

```

        +Q(j)*(-u((k-1)+j+1)+u(j-(k-1)+1)));
    else
        sum3=sum3+(P(j+3)*(2*u(j-(k-1)+1)+u((k-1)+j+1))...
        +Q(j)*(v((k-1)+j+1)));
        sum6=sum6+(P(j+3)*(v((k-1)+j+1))...
        +Q(j)*(-u((k-1)+j+1)+2*u(j-(k-1)+1)));
    end
end
end
end
KD(k)=0.25*(sum1+sum2+sum3);KE(k)=0.25*(sum4+sum5+sum6);
end
end
for k=1:(2*M+1)
    if k==1
        BG(k)=-(5/32)*((P(1))^2)-P(1)*P(2)-((P(2))^3)/P(1)-(3/4)*((P(2))^2);
    elseif k==2
        BG(k)=(3/4)*P(1)*P(2)+((P(2))^3)/P(1);
    elseif k==3
        BG(k)=(1/8)*((P(1))^2)+(3/4)*((P(2))^2);
    elseif k==4
        BG(k)=(1/4)*P(1)*P(2);
    elseif k==5
        BG(k)=(1/32)*((P(1))^2);
    else
        BG(k)=0;
    end
end
end
for k=1:(M+1)
    alpha(k,1)=GR(k);
end
for k=1:(M+1)
    beta(k,1)=BR(k);
end
for k=1:(2*M+1)
    if k==1
        sum1_1=0;sum1_2=0;sum2_1=0;sum2_2=0;sum3_1=0;sum3_2=0;
        for j=0:M
            if j==0

```

```

        sum1_1=sum1_1+0.5*P(j+3)*(2*GA(j+1));
        sum2_1=sum2_1+0.5*P(j+3)*(2*BM(j+1));
        sum3_1=sum3_1+0.5*P(j+3)*(2*BE(j+1));
    else
        sum1_2=sum1_2+0.5*P(j+3)*(GA(j+1));
        sum2_2=sum2_2+0.5*P(j+3)*(BM(j+1));
        sum3_2=sum3_2+0.5*P(j+3)*(BE(j+1));
    end
end
gamma(1,k)=sum1_1+sum1_2;gamma(2,k)=sum2_1+sum2_2;
gamma(3,k)=sum3_1+sum3_2;
delta(1,k)=0;delta(2,k)=0;delta(3,k)=0;
else
sum1_1=0;sum1_2=0;sum1_3=0;sum1_4=0;sum1_5=0;sum1_6=0;
sum2_1=0;sum2_2=0;sum2_3=0;sum2_4=0;sum2_5=0;sum2_6=0;
sum3_1=0;sum3_2=0;sum3_3=0;sum3_4=0;sum3_5=0;sum3_6=0;
for j=0:M
    if (k-1)-j>0
        sum1_1=sum1_1+(0.5)*P(j+3)*(GA((k-1)-j+1)+GA((k-1)+j+1));
        sum2_1=sum2_1+(0.5)*P(j+3)*(BM((k-1)-j+1)+BM((k-1)+j+1));
        sum3_1=sum3_1+(0.5)*P(j+3)*(BE((k-1)-j+1)+BE((k-1)+j+1));
    elseif (k-1)-j<0
        sum1_2=sum1_2+(0.5)*P(j+3)*(GA(j-(k-1)+1)+GA((k-1)+j+1));
        sum2_2=sum2_2+(0.5)*P(j+3)*(BM(j-(k-1)+1)+BM((k-1)+j+1));
        sum3_2=sum3_2+(0.5)*P(j+3)*(BE(j-(k-1)+1)+BE((k-1)+j+1));
    else
        sum1_3=sum1_3+(0.5)*P(j+3)*(2*GA(j-(k-1)+1)+GA((k-1)+j+1));
        sum2_3=sum2_3+(0.5)*P(j+3)*(2*BM(j-(k-1)+1)+BM((k-1)+j+1));
        sum3_3=sum3_3+(0.5)*P(j+3)*(2*BE(j-(k-1)+1)+BE((k-1)+j+1));
    end
end
end
gamma(1,k)=sum1_1+sum1_2+sum1_3;gamma(2,k)=sum2_1+sum2_2+sum2_3;
gamma(3,k)=sum3_1+sum3_2+sum3_3;
for j=1:M
    if (k-1)-j>0
        sum1_4=sum1_4+(0.5)*Q(j)*(GA((k-1)-j+1)-GA((k-1)+j+1));
        sum2_4=sum2_4+(0.5)*Q(j)*(BM((k-1)-j+1)-BM((k-1)+j+1));
        sum3_4=sum3_4+(0.5)*Q(j)*(BE((k-1)-j+1)-BE((k-1)+j+1));
    end
end

```

```

elseif (k-1)-j<0
    sum1_5=sum1_5+(0.5)*Q(j)*(GA(j-(k-1)+1)-GA((k-1)+j+1));
    sum2_5=sum2_5+(0.5)*Q(j)*(BM(j-(k-1)+1)-BM((k-1)+j+1));
    sum3_5=sum3_5+(0.5)*Q(j)*(BE(j-(k-1)+1)-BE((k-1)+j+1));
else
    sum1_6=sum1_6+(0.5)*Q(j)*(2*GA(j-(k-1)+1)-GA((k-1)+j+1));
    sum2_6=sum2_6+(0.5)*Q(j)*(2*BM(j-(k-1)+1)-BM((k-1)+j+1));
    sum3_6=sum3_6+(0.5)*Q(j)*(2*BE(j-(k-1)+1)-BE((k-1)+j+1));
end
end
delta(1,k)=sum1_4+sum1_5+sum1_6;delta(2,k)=sum2_4+sum2_5+sum2_6;
delta(3,k)=sum3_4+sum3_5+sum3_6;
end
end
for k=1:(2*M+1)
    gamma(4,k)=GA(k);delta(4,k)=0;
end
for k=1:(2*M+1)
    zeta(1,k)=u(k);eta(1,k)=v(k);
end
for k=1:(2*M+1)
    zeta(2,k)=KD(k)+BG(k);eta(2,k)=KE(k);
end
for i=1:(2*M+3)
    if i==1
        sum=0;
        for k=1:M
            sum=sum+(1/k)*delta(1,k+1);
        end
        R(i,1)=-zeta(2,1)+lambda*sum;
    elseif i==(2*M+2)
        sum1=0;sum2=0;
        for k=0:M
            sum1=sum1+((-1)^k)*BG(k+1);
        end
        for k=1:M
            sum2=sum2+((1-((-1)^k))/k)*delta(1,k+1);
        end
    end
end

```

```

        R(i,1)=sum1-lambda*sum2;
elseif i==(2*M+3)
    R(i,1)=-gamma(1,1);
elseif i>=2 & i<=M+1
    R(i,1)=-zeta(2,i)-(lambda/(i-1))*delta(1,i);
else
    R(i,1)=-eta(2,i-M)+(lambda/(i-M-1))*gamma(1,i-M);
end
end
if norm(R,inf)<1e-8
    break
end
for j=1:(2*M+3)
    for i=1:(2*M+3)
        if j==1
            if i==1
                sum=0;
                for k=1:M
                    sum=sum+(1/k)*delta(2,k+1);
                end
                A(i,j)=alpha(1,1)-lambda*sum;
            elseif i==(2*M+2)
                sum1=0;sum2=0;
                for k=0:M
                    sum1=sum1+((-1)^(k+1))*alpha(k+1,1);
                end
                for k=1:M
                    sum2=sum2+(1/k)*(1-(-1)^(k))*delta(2,k+1);
                end
                A(i,j)=sum1+lambda*sum2;
            elseif i==(2*M+3)
                A(i,j)=gamma(2,1);
            elseif i>=2 & i<=M+1
                A(i,j)=alpha(i,1)+(lambda/(i-1))*delta(2,i);
            else
                A(i,j)=-lambda/(i-M-1)*gamma(2,i-M);
            end
        elseif j==2

```

```

if i==1
    sum=0;
    for k=1:M
        sum=sum+(1/k)*delta(3,k+1);
    end
    A(i,j)=beta(1,1)-lambda*sum;
elseif i==(2*M+2)
    sum1=0;sum2=0;
    for k=0:M
        sum1=sum1+((-1)^(k+1))*beta(k+1,1);
    end
    for k=1:M
        sum2=sum2+(1/k)*(1-(-1)^k)*delta(3,k+1);
    end
    A(i,j)=sum1+lambda*sum2;
elseif i==(2*M+3)
    A(i,j)=gamma(3,1);
elseif i>=2 & i<=M+1
    A(i,j)=beta(i,1)+(lambda/(i-1))*delta(3,i);
else
    A(i,j)=-(lambda/(i-M-1))*gamma(3,i-M);
end
elseif j>=3 & j<=M+3
    if i==1
        sum1=0; sum2=0; sum3=0;
        for k=1:M
            if k-(j-3)>0
                sum1=sum1+(1/k)*(delta(4,k-(j-3)+1)+delta(4,(j-3)+k+1));
            elseif k-(j-3)<0
                sum2=sum2+(1/k)*(-delta(4,(j-3)-k+1)+delta(4,(j-3)+k+1));
            else
                sum3=sum3+(1/k)*(delta(4,(j-3)+k+1));
            end
        end
    end
    if j==3
        A(i,j)=(1/2)*(2*zeta(1,(j-3)+1))-(lambda/2)*(sum1+sum2+sum3);
    else
        A(i,j)=(1/2)*(zeta(1,(j-3)+1))-(lambda/2)*(sum1+sum2+sum3);
    end
end

```

```

end
elseif i==(2*M+2)
    sum1=0; sum2=0; sum3=0;
    for k=1:M
        if k-(j-3)>0
            sum1=sum1+((1-((-1)^k))/k)*(delta(4,k-(j-3)+1)...
                +delta(4,(j-3)+k+1));
        elseif k-(j-3)<0
            sum2=sum2+((1-((-1)^k))/k)*(-delta(4,(j-3)-k+1)...
                +delta(4,(j-3)+k+1));
        else
            sum3=sum3+((1-((-1)^k))/k)*(delta(4,(j-3)+k+1));
        end
    end
    end
    A(i,j)=(lambda/2)*(sum1+sum2+sum3);
elseif i==(2*M+3)
    if j==3
        A(i,j)=(1/2)*(2*gamma(4,(j-3)+1));
    else
        A(i,j)=(1/2)*(gamma(4,(j-3)+1));
    end
end
elseif i>=2 & i<=M+1
    if (i-1)-(j-3)>0
        A(i,j)=(1/2)*(zeta(1,(i-1)-(j-3)+1)+zeta(1,(i-1)+(j-3)+1))...
            +(lambda/(2*(i-1)))*(delta(4,(i-1)-(j-3)+1)...
                +delta(4,(i-1)+(j-3)+1));
    elseif (i-1)-(j-3)<0
        A(i,j)=(1/2)*(zeta(1,(j-3)-(i-1)+1)+zeta(1,(i-1)+(j-3)+1))...
            +(lambda/(2*(i-1)))*(-delta(4,(j-3)-(i-1)+1)...
                +delta(4,(i-1)+(j-3)+1));
    else
        A(i,j)=(1/2)*(2*zeta(1,(j-3)-(i-1)+1)+zeta(1,(i-1)+(j-3)+1))...
            +(lambda/(2*(i-1)))*(delta(4,(i-1)+(j-3)+1));
    end
end
else
    if (i-M-1)-(j-3)>0
        A(i,j)=(1/2)*(eta(1,(i-M-1)-(j-3)+1)+eta(1,(i-M-1)+(j-3)+1))...
            -(lambda/(2*(i-M-1)))*(gamma(4,(i-M-1)-(j-3)+1)...

```

```

        +gamma(4,(i-M-1)+(j-3)+1));
elseif (i-M-1)-(j-3)<0
    A(i,j)=(1/2)*(-eta(1,(j-3)-(i-M-1)+1)+eta(1,(i-M-1)+(j-3)+1))...
        -(lambda/(2*(i-M-1)))*(gamma(4,(j-3)-(i-M-1)+1)...
        +gamma(4,(i-M-1)+(j-3)+1));
else
    A(i,j)=(1/2)*(eta(1,(i-M-1)+(j-3)+1))...
        -(lambda/(2*(i-M-1)))*(2*gamma(4,(j-3)-(i-M-1)+1)...
        +gamma(4,(i-M-1)+(j-3)+1));
end
end
else
    if i==1
        sum1=0; sum2=0; sum3=0;
        for k=1:M
            if k-(j-M-3)>0
                sum1=sum1+(1/k)*(gamma(4,k-(j-M-3)+1)-gamma(4,(j-M-3)+k+1));
            elseif k-(j-M-3)<0
                sum2=sum2+(1/k)*(gamma(4,(j-M-3)-k+1)-gamma(4,(j-M-3)+k+1));
            else
                sum3=sum3+(1/k)*(2*gamma(4,(j-M-3)-k+1)...
                    -gamma(4,(j-M-3)+k+1));
            end
        end
        A(i,j)=(1/2)*(eta(1,(j-M-3)+1))-(lambda/2)*(sum1+sum2+sum3);
    elseif i==(2*M+2)
        sum1=0; sum2=0; sum3=0;
        for k=1:M
            if k-(j-M-3)>0
                sum1=sum1+((1-((-1)^k))/k)*(gamma(4,k-(j-M-3)+1)...
                    -gamma(4,(j-M-3)+k+1));
            elseif k-(j-M-3)<0
                sum2=sum2+((1-((-1)^k))/k)*(gamma(4,(j-M-3)-k+1)...
                    -gamma(4,(j-M-3)+k+1));
            else
                sum3=sum3+((1-((-1)^k))/k)*(2*gamma(4,(j-M-3)-k+1)...
                    -gamma(4,(j-M-3)+k+1));
            end
        end
    end
end

```

```

end
A(i,j)=(lambda/2)*(sum1+sum2+sum3);
elseif i==(2*M+3)
A(i,j)=(1/2)*(delta(4,(j-M-3)+1));
elseif i>=2 &i<=M+1
if (i-1)-(j-M-3)>0
A(i,j)=(1/2)*(-eta(1,(i-1)-(j-M-3)+1)+eta(1,(i-1)+(j-M-3)+1))...
+(lambda/(2*(i-1)))*(gamma(4,(i-1)-(j-M-3)+1)...
-gamma(4,(i-1)+(j-M-3)+1));
elseif (i-1)-(j-M-3)<0
A(i,j)=(1/2)*(eta(1,(j-M-3)-(i-1)+1)+eta(1,(i-1)+(j-M-3)+1))...
+(lambda/(2*(i-1)))*(gamma(4,(j-M-3)-(i-1)+1)...
-gamma(4,(i-1)+(j-M-3)+1));
else
A(i,j)=(1/2)*(eta(1,(i-1)+(j-M-3)+1))...
+(lambda/(2*(i-1)))*(2*gamma(4,(j-M-3)-(i-1)+1)...
-gamma(4,(i-1)+(j-M-3)+1));
end
end
else
if (i-M-1)-(j-M-3)>0
A(i,j)=(1/2)*(zeta(1,(i-M-1)-(j-M-3)+1)-zeta(1,(i-M-1)+(j-M-3)+1))...
-(lambda/(2*(i-M-1)))*(-delta(4,(i-M-1)-(j-M-3)+1)...
+delta(4,(i-M-1)+(j-M-3)+1));
elseif (i-M-1)-(j-M-3)<0
A(i,j)=(1/2)*(zeta(1,(j-M-3)-(i-M-1)+1)-zeta(1,(i-M-1)+(j-M-3)+1))...
-(lambda/(2*(i-M-1)))*(delta(4,(j-M-3)-(i-M-1)+1)...
+delta(4,(i-M-1)+(j-M-3)+1));
else
A(i,j)=(1/2)*(2*zeta(1,(j-M-3)-(i-M-1)+1)-zeta(1,(i-M-1)+(j-M-3)+1))...
-(lambda/(2*(i-M-1)))*(delta(4,(i-M-1)+(j-M-3)+1));
end
end
end
end
end
if rank(A)~=2*M+3
break
end

```

```

X=A\R;
for k=1:(M+3)
    P(k)=P(k)+X(k,1);
end
for k=1:M
    Q(k)=Q(k)+X(k+(M+3),1);
end
end
end
phi=0:0.1:2*pi; sum1=0; sum2=0;
for j=0:M
    if j==0
        sum1=sum1+0.5*P(3)*sin(phi); sum2=sum2+(0.5)*P(3)*sin(phi);
    else
        sum1=sum1+(-0.5)*Q(j)*cos((j+1)*phi)+0.5*P(j+3)*sin((j+1)*phi);
        sum2=sum2+(0.5)*Q(j)*cos((j-1)*phi)-0.5*P(j+3)*sin((j-1)*phi);
    end
end
end
e=P(1)*cos(phi)+P(2); r=-P(1)*(sum1+sum2);
tspan=[-20 20]; y0=[(P(1)+P(2)),0]; [t,y]=ode45(@odeq,tspan,y0);
plot(y(:,1),y(:,2),'g',e,r,'kx')
xlabel('x'); ylabel('xdot');

```