Frequency response of nonlinear oscillator

H.G.E. Meijer

June 23, 2023

1 A nonlinear periodically forced oscillator (Duffing)

In physics and engineering, we often look at the frequency response of an oscillator, i.e., the amplitude of the oscillation evoked by periodic forcing. As an example, we consider the Duffing oscillator given by the following second-order equation

$$mx'' + \delta x' + \alpha x + \beta x^3 = \gamma \cos(\omega t), \tag{1}$$

where δ is the damping coefficient, α the stiffnes constant, and a nonlinear correction β. We set the mass $m = 1$ as it would only scale the other constants. The forcing has an amplitude γ and a frequency ω. Without forcing the solution $x(t)$ will decay to zero. When applying forcing with a small amplitude, the result may be a periodic response. This tutorial aims to show how to obtain the amplitude using MatCont [3]. This toolbox used numerical continuation to track solutions as a parameter varies. The periodic response is numerically approximated using orthogonal collocation, i.e. a discrete mesh with approximating polynomials satisfying the system on each interval glued together at the mesh points. Any other continuation package allowing for tracking periodic orbits can be used too. Here, we focus on this toolbox. This tutorial has been tested with MATCONT version 7p2 and MATLABR2020A.

1.1 Modelling

To interpret these solutions, we review a derivation for a nonlinear spring. Starting from Newton's second law, we have

$$mx'' = F_{\text{spring}} + F_{\text{friction}} + F_{\text{forcing}}.$$

According to Hooke's law, the force generated by the spring is proportional to the deviation x from rest, i.e., $F_{\text{spring}} = -\alpha x$ and tries to restore the spring to its rest position. The elasticity is no longer linear for large deviations, and a nonlinear term βx^3 must be included. The quadratic term vanishes as we require the force to be an odd function of position x. The constant β is a material property, and the spring is hardening for $\beta < 0$ as the restoring force becomes larger, and softening for $\beta > 0$. Alternatively, thinking of the mathematical pendulum with gravity instead of a spring force, we expand the force including one more term in the Taylor expansion, i.e., $F_{\text{gravity}} = -mg\sin(x) \approx -mg(x - \frac{1}{6}x^3 + ...)$. It really is a better approximation, as there are still two saddle equilibria. As with the ideal pendulum, the system also permits a constant of motion, sometimes used for analysis. Next, friction is modelled as a force proportional to the speed x' and in the opposite direction. Finally, periodic forcing models the dominant Fourier mode of a general forcing signal.

1.2 The augmented nonlinear system

MATCONT allows continuation of periodic solutions of autonomous nonlinear first-order equations. As we have a second-order nonautonomous equation (1), we do some transformations. First, we introduce the dummy variable $y = x'$ for the speed. We then obtain the first-order nonautonomous system

$$\left\{ \begin{array}{rcl} x' &=& y, \\ x'' = y' &=& \gamma\cos(\omega t) - \alpha x - \delta y - \beta x^3. \end{array} \right. \tag{2}$$

This system is still nonautonomous. To overcome this, we use the Hopf normal form [1]

$$\left\{\begin{array}{rcl} u' &=& -\omega v + u(1-u^2-v^2), \\ v' &=& \omega u + v(1-u^2-v^2). \end{array}\right. \tag{3}$$

The solutions of this system follow a periodic orbit on the circle of radius 1. For more details, see the exercises below. They transverse this circle with frequency ω. The initial conditions set the phase. Choosing $(u_0, v_0) = (1, 0)$, we have $u(t) = \cos(\omega t)$. Combining (2) and (3), we arrive at a system that we can use for numerical continuation

$$\left\{\begin{array}{rcl} x' &=& y, \\ y' &=& \gamma u - \alpha x - \delta y - \beta x^3, \\ u' &=& -\omega v + u(1-u^2-v^2), \\ v' &=& \omega u + v(1-u^2-v^2). \end{array}\right. \tag{4}$$

2 MatCont Steps to Obtain Frequency Response

We assume you have downloaded the textscMatCont package from SourceForge, and extracted the package into some folder. Change your Matlab working folder to this folder, i.e. the folder containing the file matcont.m.

2.1 Setting up the System

Type "matcont" from the Matlab command line to start up the toolbox. The main window appears, and possibly others too, if some other system is already active. We want the system (4) to be available in MatCont, and to define this system, we choose in the main window **Select** $\rightarrow$ **System** $\rightarrow$ **New**, see Figure 1(left). We then fill in all the fields according to Figure 1(right), or see the listing below for all fields. Note you can also copy-paste this input from `https://wwwhome.ewi.utwente.nl/~meijerhge/MT.txt`.

```
Name         DuffingForced
Coordinates x, y, u, v
Parameters alpha, beta, gamma, delta, omega
Equations
x'= y
y'= gamma*u- alpha*x -delta *y -beta*x^3
u'= -omega*v+u*(1 -u^2 -v^2)
v'= omega*u+v*(1 -u^2 -v^2)
```

If you have access to the Symbolic Toolbox in Matlab, then select symbolic derivatives up to order three. If not, textscMatCont will use finite differences to determine certain derivatives. When you are ready, press "OK".

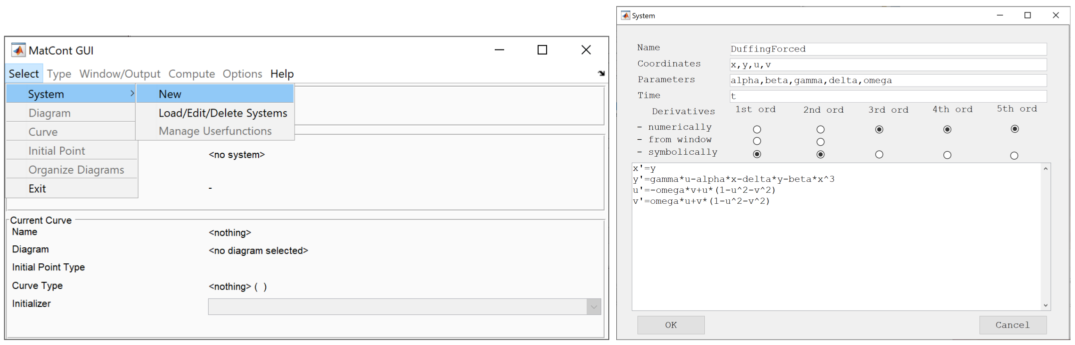
2.2 Initial Simulation on an Oscillation

We will now perform a simulation in order to simulate roughly one period of the periodic orbit. This simulation will serve as initial data for the continuation. Selecting a proper orbit segment from an arbitrary simulation is difficult. So we first do one long simulation and then pick up the final point of this simulation to simulate one forcing period. This procedure may seem like doing work twice, but it is the most robust method to provide the initial data.

We set the *Type* of the Initial Point to *Point*, see Figure 2(top). The Curve Type is now set to *Orbit* automatically. Two windows appear, called Starter and Integrator.[2] We can enter initial conditions and

[1] Another idea would be to introduce a cyclic variable $\phi = \omega t$ satisfying $\phi' = \omega$, but the continuation of periodic orbits of systems with cyclic variables is (currently) not supported in MatCont. After transforming to polar coordinates, the system reads $r' = r(1-r^2), \theta' = \omega$, with a fixed point $r^* = 1$ So, ignoring transients, the solution for u approaches $u = r^* \cos(\omega t + \theta_0)$, and as this part is autonomous we may consider $\theta_0 = 0$.

[2] If for some reason they do not appear, then right-click in the lower (curve) part of the main MatCont window, and select them in the Pop-up menu.



	<no system>
	-
Current Curve Name	<nothing>
Diagram	<no diagram selected>
Initial Point Type	
Curve Type	<nothing> ( )
Initializer	

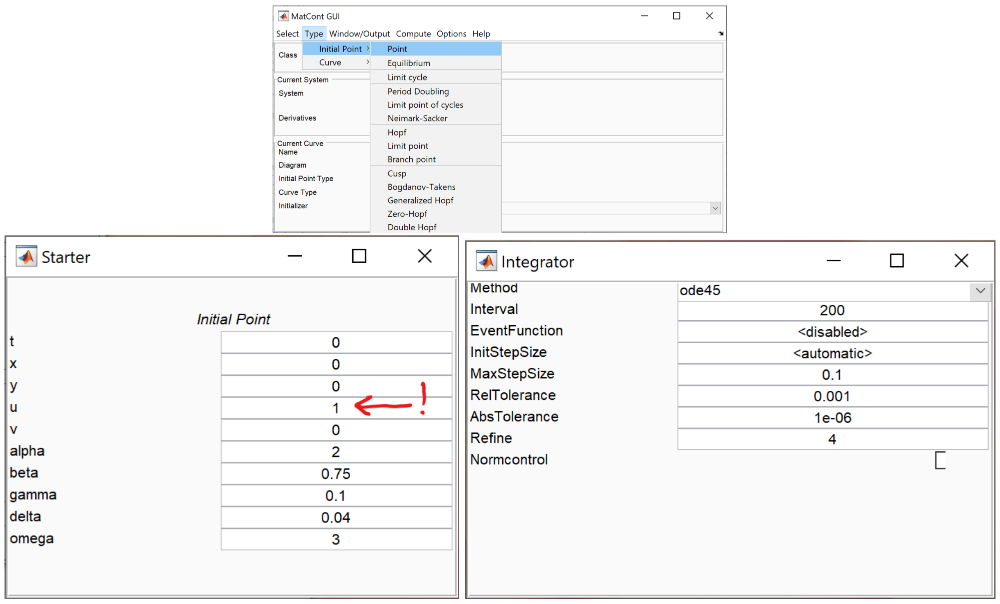
Name	DuffingForced
Coordinates	x,y,u,v
Parameters	alpha,beta,gamma,delta,omega
Time	t

Derivatives	1st ord	2nd ord	3rd ord	4th ord	5th ord
- numerically	○	○	●	●	●
- from window	○	○			
- symbolically	●	●	○	○	○

```
x'=y
y'=gamma*u-alpha*x-delta*y-beta*x^3
u'=-omega*v+u*(1-u^2-v^2)
v'=omega*u+v*(1-u^2-v^2)
```

Figure 1: Left: The main MatCont window starting from some arbitrary system. Right: Entering the equations for the Morris-Lecar model.

parameter values. We choose $\alpha = 2, \beta = 0.2, \gamma = 0.4, \delta = 0.04, \omega = 1$. Note the natural frequency of this oscillator, based on the linear terms, is $\Omega = \sqrt{\alpha} \approx 1.41$. For the initial condition, we choose $x = y = 0$ as this will not matter too much. For the auxiliary variables, we set $u = 1$ and $v = 0$ to start on the attracting periodic orbit in this subsystem. Starting with $u = v = 0$ will mean they stay at the origin, and then there is no periodic forcing. So any other nonzero initial condition can be chosen, but then transients may last longer. For the integrator settings, we set *Interval=200* and *MaxStepSize=0.1*. The first option ensures we simulate long enough to achieve convergence to the periodic response after transients, and the latter gives sufficient accuracy. See also Figure 2 for all fields. It is time to perform

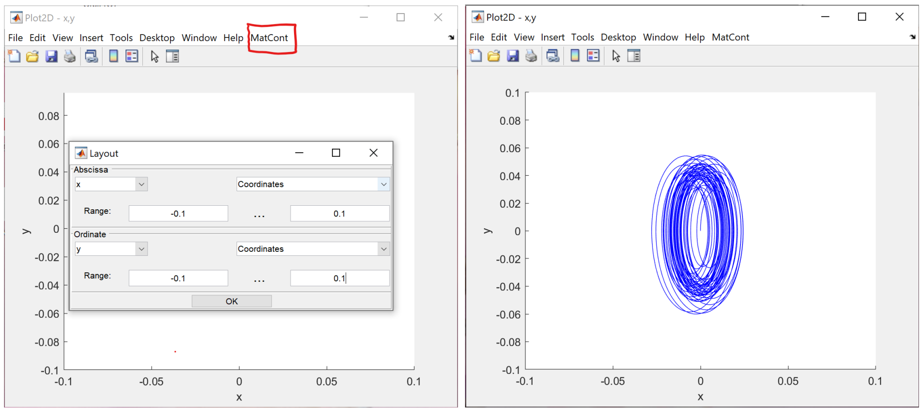


	Initial Point
t	0
x	0
y	0
u	1
v	0
alpha	2
beta	0.75
gamma	0.1
delta	0.04
omega	3

Method	ode45
Interval	200
EventFunction	<disabled>
InitStepSize	<automatic>
MaxStepSize	0.1
RelTolerance	0.001
AbsTolerance	1e-06
Refine	4
Normcontrol	☐

Figure 2: Top: Selecting the type for the initial point. Bottom: Numerical values and Integrator settings.

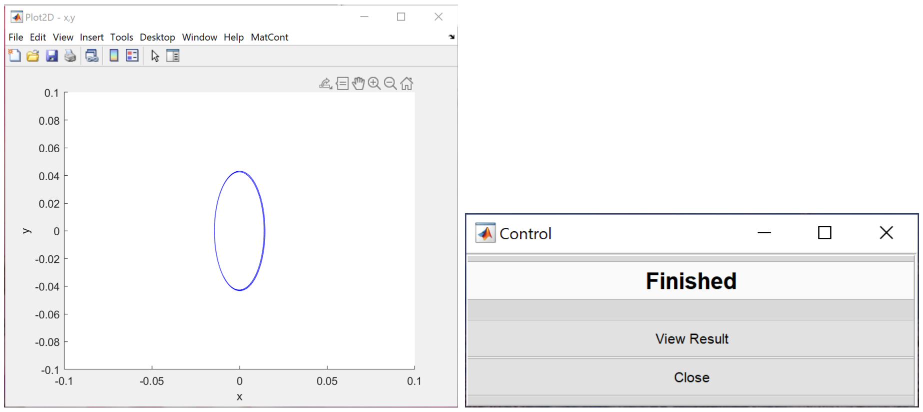
the simulation; Press "Compute|Forward" in the main window. Quickly, a pop-up window with messages appears, and the simulation has finished. Now that we have performed the simulation, we will visualize the orbit in the xy-phase plane via "Window/Output|Graphic|2D plot". A new window appears, and by clicking on "MatCont|Layout" we can set the Axis properties. Select "MatCont|Redraw Curve". The result looks a bit cluttered as initial transients are also plotted, see Figure 3. We could also have opened the plot window first before simulating, but plotting during simulations is rather slow, especially if we want transients to fade out. To see the periodic orbit, we first clear the window "2DPlot|MatCont|Clear", and then in the Main window,



Abscissa
x — Coordinates
Range: -0.1 ... 0.1
Ordinate
y — Coordinates
Range: -0.1 ... 0.1

Figure 3: Left: Setting the Layout for the 2D Plot. Right: The simulated orbit projected onto the xy-plane with transients towards the periodic orbit.

we select "Compute|Extend" to extend the current simulation. The result looks as in Figure 4. You can also check in the uv-plane that the uv-subsystem traverses the unit circle by selecting different variables to plot in the Layout.



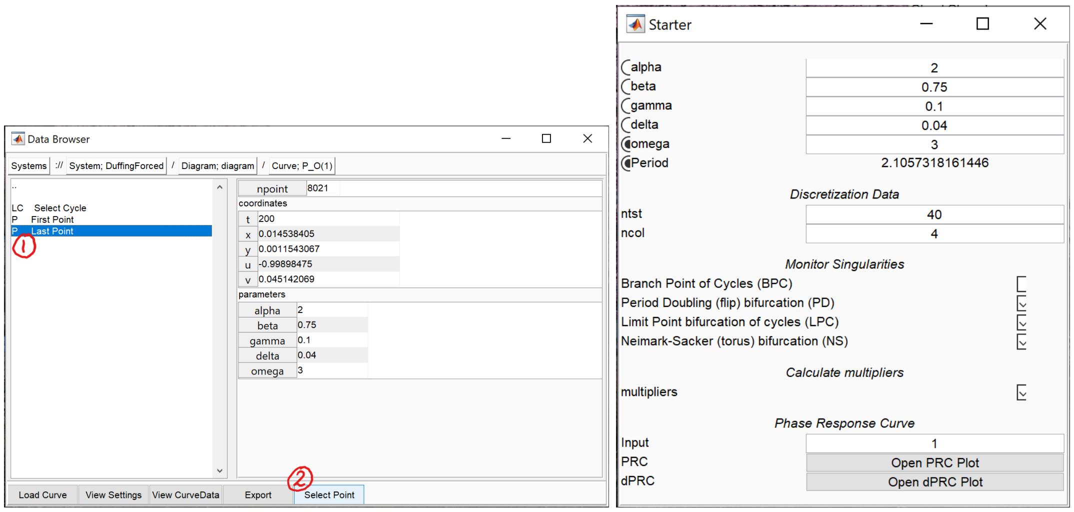
Finished

Figure 4: Left: zooming in on the orbit by extending after transients. Right: Message window when the simulation has finished.

We can now gather the proper initial data. After pressing "View Result" in the Control/Message window, a window called *Data Browser* opens. On the left, click on "P Last Point", and next "Select Point" on the Bottom, see Figure 5. In the Integrator Window set *Interval=2.2*, which is a little more than one period of the forcing ($2\pi/\omega \approx 2.09$). We start the simulation with "Compute|Forward" in the Main window, and in the Message window, we click on "View Result". Now in the Data Browser, LC Select Cycle is highlighted, and we select that as new initial data; Press *Select Point*. A small window appears asking you to set the tolerance and the number of intervals. These standard settings are ok. Press "OK", and we are ready to proceed to the continuation.

2.3 Continuation of the Limit Cycle

Here we will perform the numerical continuation of the limit cycle, i.e. an isolated periodic orbit. The continuation requires selecting two parameters; we select the forcing frequency ω and the period, see Figure 5(right), by ticking the boxes. You can observe that the period equals $2\pi/\omega$, reflecting the periodic response. We turn off the detection of BPC-points. The standard settings in the Continuer window are fine. To prepare



Data Browser

Systems :// System; DuffingForced / Diagram; diagram / Curve; P_O(1)

..
LC Select Cycle
P First Point
P Last Point

npoint	8021
coordinates	
t	200
x	0.014538405
y	0.0011543067
u	-0.99898475
v	0.045142069
parameters	
alpha	2
beta	0.75
gamma	0.1
delta	0.04
omega	3

Starter

alpha	2
beta	0.75
gamma	0.1
delta	0.04
omega	3
Period	2.1057318161446

Discretization Data

ntst	40
ncol	4

Monitor Singularities

Branch Point of Cycles (BPC) [ ]
Period Doubling (flip) bifurcation (PD) [✓]
Limit Point bifurcation of cycles (LPC) [✓]
Neimark-Sacker (torus) bifurcation (NS) [✓]

Calculate multipliers

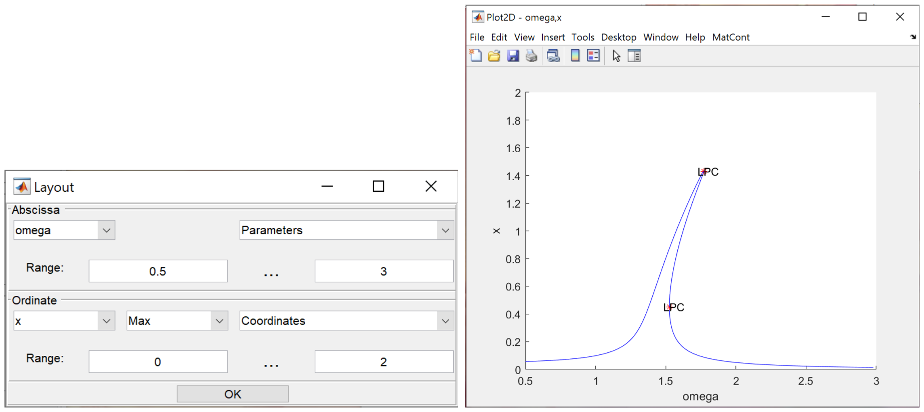
multipliers [✓]

Phase Response Curve

Input	1
PRC	Open PRC Plot
dPRC	Open dPRC Plot

Figure 5: Left:Selecting the final point of a simulation as new initial point. Right: Preparing the continuation of the limit cycle.

for the output, we change the layout of the 2D plot. The amplitude really is the maximal absolute value of the variable x along the limit cycle. Therefore we select the parameter ω on the horizontal axis (Abscissa) and the Max of x on the vertical one (Ordinate), see Figure 6(Left). Now select ”Compute|Forward”. If all is well, the parameter ω will decrease during continuation initially, otherwise select ”Compute|Backward”. You will get a message **LPC** twice, just press ”Resume” and extend the computation until the parameter ω has decreased to 0.5. You will now have the plot in Figure 6(right). This plot, in principle, is the frequency response plot. Note the large response near the natural frequency.



Layout

Abscissa

omega | Parameters

Range: 0.5 ... 3

Ordinate

x | Max | Coordinates

Range: 0 ... 2

OK

Figure 6: Setting the Layout for the 2D-Plot and the result of the continuation.

2.4 Final Visualization of the Frequency Response

We can improve the frequency plot by indicating the stability of the branches. In the 2D Plot window, we select *MatCont|Plot Properties*. Search for the field *LC if unstable* and enter: ’Color’, ’red’, ’linestyle’, ’:’ . The dashed line style (’- -’) did not work for this tutorial.

The two LPC points explain the hysteresis phenomenon occurring when ω is slowly varied. Starting with $\omega = 1.0$ and increasing it slowly, we track the stable branch of the limit cycle until we get close to the upper

LPC near $\omega = 1.77$. Continuing further, the amplitude suddenly drops to the other stable branch. Similarly, starting with $\omega = 2.0$ and decreasing the parameter value, the amplitude suddenly jumps up once we are past the LPC at $\omega = 1.52$.

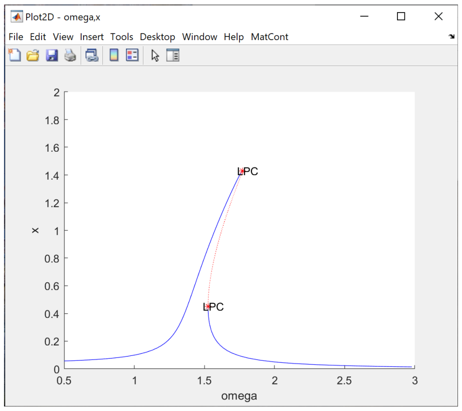


Figure 7: The frequency response curve for a hardening nonlinearity ($\beta > 0$) with the tip bending to the right. Changing the plot options, the unstable part is highlighted.

3 Remarks and Exercises

1. In the Appendix below, there are two m-files to achieve the same using the Matlab command line instead of the GUI.

2. For small amplitudes γ, a nonlinear equation for the frequency response can be derived.

   - Assume $x(t) = a\cos(\omega t) + b\sin(\omega t)$. Insert this into equation (1) and collect the coefficients of the terms with $\cos(\omega t)$ and $\sin(\omega t)$. Ignore the 3ω-terms.
   - Combine the equations given by these coefficients to obtain an equation $F(z, \omega)$ for the amplitude $z^2 = a^2 + b^2$.
     Hint: Square and add both.
   - Verify the accuracy of this approximation for strong forcing, i.e. high values for γ, by comparing to the frequency response obtained using numerical continuation. Set $\alpha = 1, \delta = 0.2, \beta = 0.4$ and $\gamma = 0.8$.
     Hint: the analytical approximation is a sixth-order polynomial in the amplitude variable that requires numerical approximations. Roots of this equation can be found using numerical continuation too, using the system $z' = F(z, \omega)$ with ω as a parameter.

3. Obtain the frequency response for the forced van der Pol oscillator

$$m\ddot{x} + \alpha x + \beta\dot{x}(1 - x^2) = \gamma\cos(\omega t).$$

4 A classical approximation technique for comparison

Traditionally, this equation has been studied using the method of multiple scales [4]. Here, we rewrite the equations of motion (1) as a perturbed oscillator as follows,

$$mx'' + \omega_n^2 x = \varepsilon\left(\tilde{F}\cos(\omega t) - \tilde{\delta}x' - \tilde{\beta}x^3\right) \tag{5}$$

with natural frequency $\omega_n := \sqrt{\alpha}$ and a small parameter ε. We use the $\tilde{}$ to indicate the rescaled version of the parameter. We now look for a periodic solution that is a modulation of a pure harmonic oscillation using an expansion of the form

$$x(t) = x_0(T_0, T_1, \dots) + \varepsilon x_1(T_0, T_1, \dots) + \dots, \qquad T_n = \varepsilon^n t, \quad n = 0, 1, 2, \dots. \tag{6}$$

The use of T_1 and x_1 indicates the method of multiple scales. This method yields equations that are solved successively and leads to a formula for the amplitude that is valid for small values of ε. Substituting (6) into (5) and collecting powers of ε, we get

$$(D_0^2 + \omega_n^2)x_0 = 0, \tag{7a}$$
$$(D_0^2 + \omega_n^2)x_1 = \tilde{F}\cos(\omega T_0) - 2D_0D_1x_0 - \tilde{\delta}D_0x_0 - \tilde{\beta}x_0^3, \tag{7b}$$

where $D_i^j = \frac{\partial^j}{\partial T_i^j}$ indicates the partial derivative w.r.t. to the ith-timescale. The general solution for Eq. (7a) is given by

$$x_0 = A(T_1)e^{i\omega_n T_0} + \bar{A}(T_1)e^{-i\omega_n T_0},$$

where the amplitude A does not depend on T_0, but possibly on the timescale T_1. Even higher scales are ignored. The goal is now to find an expression for A by finding a solution for x_1. The response frequency will follow the driving frequency, but there is some detuning w.r.t. the natural frequency. Hence, we write $\omega = \omega_n + \varepsilon\sigma$ and $\omega t = \omega_n T_0 + \sigma T_1$. Substituting the general solution of x_0 into (7b), we find

$$(D_0^2 + \omega_n^2)x_1 = \left(\frac{1}{2}\tilde{F}e^{i\sigma T_1} - \omega_n i(\tilde{\delta}A + A') - 3\tilde{\beta}A^2\bar{A}\right)e^{i\omega_n T_0} - \tilde{\beta}A^3e^{3i\omega_n T_0} + c.c., \tag{8}$$

with c.c. indicating complex conjugate. For bounded solutions x_1, we require the coefficient of the $e^{i\omega_n T_0}$-term to vanish as it is resonant. This coefficient is an ODE for A w.r.t. time T_1. We will now solve for $A(T_1)$, but once we have it, then we find $x_1 = \frac{\tilde{\beta}}{8\omega_n^2}A^3e^{i\omega_n T_0}$ where the homogeneous solution may be added. Writing $A = \frac{1}{2}a(T_1)e^{i\theta(T_1)}$, and splitting real and imaginary parts, we find the following system of modulation equations

$$\left\{\begin{array}{rcl} a' & = & -da + \frac{F}{2\omega_n}\sin(\sigma T_1 - \theta), \\ a\theta' & = & \sigma a - \frac{3\bar{\beta}a^3}{8\omega_n^2} + \frac{\bar{F}}{2\omega_n}\cos(\sigma T_1 - \theta). \end{array}\right. \tag{9}$$

A periodic solution of (5) corresponds to a stationary solution of (9). Such a solution is implicitly defined by the equation

$$\sigma = \frac{3\bar{\beta}a^2}{8\omega_n} \pm \sqrt{\left(\frac{\bar{F}}{2a\omega_n}\right)^2 - \bar{\delta}^2}.$$

Finally, given this expression, we can solve for the driving frequency ω admitting a given amplitude a. We note there is a finite interval for a to find real solutions for ω. In figure (**??**), we show the amplitude obtained by the approximation with multiple scales and the numerical approach. For small values of ε, the results agree very well, while for larger values, the approximation starts to differ considerably. As we used only the first-order term for ε, it is not strange the validity of the approximation breaks down, which is also noted in [4]. Also, note that the multiple time-scales method does not yield the stability of this branch.

References

[1] Duffing equation, Wikipedia `https://en.wikipedia.org/wiki/Duffing_equation`

[2] Takashi Kanamaru (2008) Duffing oscillator. Scholarpedia, 3(3):6327. `https://dx.doi.org/10.4249/scholarpedia.6327`

[3] A. Dhooge, and W. Govaerts, Yu.A. Kuznetsov, H.G.E. Meijer and B. Sautois, New features of the software MatCont for bifurcation analysis of dynamical systems (2008) MCMDS, Vol. 14, No. 2, pp 147-175, available at `https://sourceforge.net/projects/matcont/`.

[4] Nayfeh, A.H., Resolving Controversies in the Application of the Method of Multiple Scales and the Generalized Method of Averaging, Nonlinear Dynamics (2005) Vol. 40, pp. 6-102, `https://doi.org/10.1007/s11071-005-3937-y`

[5] Kolan, A., Lecture on ”Classical treatment of the Duffing oscillator”, (Dec 2017)`https://www.youtube.com/watch?v=iS7PJSi1gA4`, note=Accessed 25-08-2022

5 Appendix – Command Line Script

This is a minimal system definition file to define a periodically forced Duffing oscillator.

```
% Name;  DuffingForced
function out = DuffingForced
out{2} = @fun_eval;
out{9} = [];
% ----------------------------------------------------------------------------
function dydt = fun_eval(t,kmrgd,par_alpha,par_beta,par_gamma,par_delta,par_omega2)
dydt=[kmrgd(2);
par_gamma*kmrgd(3)-par_alpha*kmrgd(1)-par_delta*kmrgd(2)-par_beta*kmrgd(1)^3;
-par_omega*kmrgd(4)+kmrgd(3)*(1-kmrgd(3)^2-kmrgd(4)^2);
par_omega*kmrgd(3)+kmrgd(4)*(1-kmrgd(3)^2-kmrgd(4)^2);];
```

This is a script to determine Frequency Response of a Duffing oscillator using numerical continuation. It is assumed that the above file (`DuffingForced.m`) can be called.

```
% Name;   Test_DuffingForced
init; % Initialize Matcont

%% Initial simulation
% We assume that we have an m-file setup for MatCont
fun=DuffingForced();

%Parameter values and initital condition
p=[2; 0.75; 0.1; 0.04; 3]; %Parameters alpha,beta,gamma,delta,omega
X0=[0; 0; 1; 0]; %Initial condition; third and fourth should not be zero
P0=num2cell(p);
%Running a first simulation for transients
[  ,y] = ode45(fun{2},[0  500],[0; 0; 1; 0],[],P0{:});

% Now we take the final point and simulate over a little more than one period:
% 2pi/omega=2.1  (and add +.1=2.2)
% To initialize the continuation it is better to have smaller steps as that
% adds to accuracy more than setting RelTol or AbsTol.
opt=odeset('MaxStep',.01);
x1 = y(end,:);
[t,y] = ode45(fun{2},[0 2.15],x1,opt,P0{:});

% Let's check the response is indeed periodic, and transients are gone.
figure(1)
plot(y(:,1),y(:,2));
%% Numerical continuation of the periodic orbit
% We provide the simulated response (y) to the initializer
% In the part below, you may need to switch Backward from 1 to 0, this is
% simply unpredictable across systems.
tolerance=1e-3;
ap=5;    % omega will be the active parameter
[x0,v0]=initOrbLC(@DuffingForced,t,y,p,ap,40,4,tolerance);
opt=contset;
opt=contset(opt,'MaxNumPoints',800);
opt=contset(opt,'Backward',1);
opt=contset(opt,'Singularities',1);
opt=contset(opt,'IgnoreSingularity',[1 2 4]); %Ignore everything except LPC
```

```
[xlc1,vlc1,slc1,hlc1,flc1]=cont(@limitcycle,x0,v0,opt);
% For the backward branch, we have to re-initialize as some global
% variables change during the continuation. Else, we might get the same
% branch. We need fewer points here, and we know there are no singularities
% on this part, so we switch off monitoring for bifurcations.
[x0,v0]=initOrbLC(@DuffingForced,t,y,p,ap,40,4,tolerance);
opt=contset(opt,'MaxNumPoints',100);
opt=contset(opt,'Backward',0);
opt=contset(opt,'Singularities',0);
[xlc2,vlc2,slc2,hlc2,flc2]=cont(@limitcycle,x0,v0,opt);

%% Visualization of Results
% Determine the amplitude
A1=max(xlc1(1:4:end-2,:));
A2=max(xlc2(1:4:end-2,:));

% A simple bifurcation diagram (amplitude as function of parameter)
figure(2);clf(2);hold on;
plot(xlc1(end,:),A1,xlc2(end,:),A2);

% A polished bifurcation diagram including stability
figure(3);clf(3);hold on;
ind1=(1:slc1(2).index); % split into separate parts
ind2=(slc1(2).index:slc1(3).index);
ind3=(slc1(3).index:slc1(4).index);
plot(xlc1(end,ind1),A1(ind1),xlc1(end,ind3),A1(ind3),xlc2(end,:),A2,'Color','blue');
plot(xlc1(end,ind2),A1(ind2),'Color','red','LineStyle','--');
xlabel('omega');ylabel('Amplitude');
xlim([0.5 4]);
```