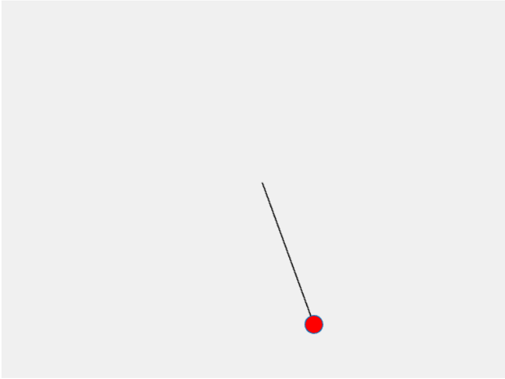
CHAOTIC BEHAVIOR IN A PLANE PENDULUM WITH A SINUSOIDAL DRIVING TORQUE AT THE PIVOT POINT

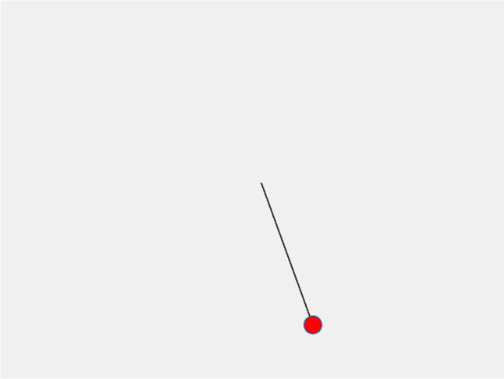
Tam Ho Won, Owen
Computational Physics

What is the problem?

- Pendulum restricted in plane motion
  - $I\frac{d^2\theta}{dt^2} + mgl\ sin\theta = 0$
  - $\frac{d^2\theta}{dt^2} = -\frac{g}{l}\theta \qquad (\theta \to small, \quad sin\theta \to \theta)$
- $\theta = Acos(\omega t) + Bsin(\omega t)$
  - ${\omega_0}^2 = \frac{g}{l}, A = \theta_{t=0}, B = \frac{\omega_{t=0}}{\omega_0}$
- Deterministic & predictable
- Periodic motion with $\mathrm{T} = \frac{2\pi}{\omega_0}$

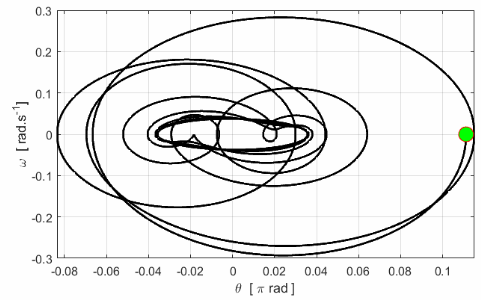


What is the problem?



- Adding driving torque?
  - $I\frac{d^2\theta}{dt^2} + b\frac{d\theta}{dt} + mgl\ sin\theta = \tau_{ext}(t) = \tau_0 \sin(\omega_{ext}t)$
- Non-linear 2nd order ODE

- Without exact analytical solution
- Approximation is available
  - Time consuming, Careful
  - Size step must be small to be meaningful



Chaos Dynamics

- Nonrandom complicated motions
  - Nonlinear or infinite-dimensional
- Sensitive to initial conditions
  - Violation of strong principle of causality
- Deterministic but practically unpredictable
- Butterfly effects

- E.g. Weather forecasting, logistic equation

Algorithm Used

- $\omega = \frac{d\theta}{dt}, \; \phi = \omega_{ext} t$
  - $\frac{d\omega}{dt} + c\omega + \frac{g}{l} sin\theta = F\sin(\omega_{ext} t)$
  - $\frac{d\theta}{dt} = \omega$
  - $\frac{d\phi}{dt} = \omega_{ext}$

- Euler's Method
- 4^{th} Order Runge-Kutta's Method

Euler's Method

- Discretize set of 1st order ODE
  - $\omega_{i+1} - \omega_i = h(-c\omega_i - \frac{g}{l} sin\theta_i + F_{ext}\sin(\omega_{ext} t_i))$
  - $\theta_{i+1} - \theta_i = h\omega$
  - $t_{i+1} - t_i = h$

```
%=====================================================================
% Equation of motion
function [h_f, h_g] = motion_fn()
    [g,l,~,~,extF,extF_w,c,~,~,~] = get_var();   %%Get initial parameters
    h_f = @(time,theta,w) w;
    h_g = @(time,theta,w) -g/l*sin(theta) - c*w + extF*sin(extF_w*(time));
end
```

```
function [time,theta,w] = euler_1(time,theta,w)
    [~,~,~,~,~,~,~,dt,~,duration] = get_var();
    [f,g] = motion_fn;
%     [f,g] = test_function_1;   %%For Function Test
    ind = 0:dt:duration;
    for i = 1:(length(ind)-1)
        %Two first order differential equations
        w(i+1) = w(i) + dt*g(time(i),theta(i),w(i));
        theta(i+1) = theta(i)+f(time(i),theta(i),w(i))*dt;
        time(i+1) = dt*i;
    end
end
```

4^{th} Order Runge-Kutta's Method (RK4)

- $\theta_{i+1} = \theta_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$
- $\omega_{i+1} = \omega_i + \frac{h}{6}(l_1 + 2l_2 + 2l_3 + l_4)$
  - $k_1 = f(time_i, \theta_i, \omega_i)$
  - $l_1 = g(time_i, \theta_i, \omega_i)$
  - $k_2 = f(time_i + \frac{h}{2}, \theta_i + \frac{hk_1}{2}, \omega_i + \frac{hl_1}{2})$
  - $l_2 = g(time_i + \frac{h}{2}, \theta_i + \frac{hk_1}{2}, \omega_i + \frac{hl_1}{2})$
  - $k_3 = f(time_i + \frac{h}{2}, \theta_i + \frac{hk_2}{2}, \omega_i + \frac{hl_2}{2})$
  - $l_3 = g(time_i + \frac{h}{2}, \theta_i + \frac{hk_2}{2}, \omega_i + \frac{hl_2}{2})$
  - $k_4 = f(time_i + h, \theta_i + hk_3, \omega_i + hk_3)$
  - $l_4 = g(time_i + h, \theta_i + hk_3, \omega_i + hk_3)$

```
%==================================================================
% Equation of motion
function [h_f, h_g] = motion_fn()
    [g,l,~,~,extF,extF_w,c,~,~,~] = get_var();    %%Get initial parameters
    h_f = @(time,theta,w) w;
    h_g = @(time,theta,w) -g/l*sin(theta) - c*w + extF*sin(extF_w*(time));
end
```

4th Order Runge-Kutta's Method (RK4)

```
function [theta_new,w_new] = RK4_funct(time,theta,w,dt)
    [f, g] = motion_fn;
    %     [f,g] = test_function_1;   %%For Function Test
    k1 = f(time,          theta,              w);
    l1 = g(time,          theta,              w);
    k2 = f(time+dt/2,     theta+k1*dt/2,      w+l1*dt/2);
    l2 = g(time+dt/2,     theta+k1*dt/2,      w+l1*dt/2);
    k3 = f(time+dt/2,     theta+k2*dt/2,      w+l2*dt/2);
    l3 = g(time+dt/2,     theta+k2*dt/2,      w+l2*dt/2);
    k4 = f(time+dt,       theta+k3*dt,        w+l3*dt);
    l4 = g(time+dt,       theta+k3*dt,        w+k3*dt);
    theta_new = theta + (k1 + 2*k2 + 2*k3 + k4)*dt/6;
    w_new = w + (l1 + 2*l2 + 2*l3 + l4)*dt/6;
end
```

```matlab
%======================================================================
% Runga-Kutta 4th order
function [time,theta,w] = RK4(time,theta,w)
    [~,~,~,~,~,~,~,dt,~,duration] = get_var();
    ind = 0:dt:duration;
    for i = 1:(length(ind)-1)
        [theta(i+1),w(i+1)]= RK4_funct(time(i),theta(i),w(i),dt);
        if i>1 %%For error analysis
            [theta_2h,~]= RK4_funct(time(i-1),theta(i-1),w(i-1),dt*2);
            abs_error_1(1,i)=theta_2h-theta(i+1);
            rel_error_1(1,i)=((theta_2h-theta(i+1))/min(theta_2h,theta(i+1)));
            if abs(abs_error_1(i))>1.e-3 || abs(rel_error_1(i))>1.e-3
                disp('Error too large');
                return
            end
        end
        time(i+1) = dt*i;
    end
    fprintf('Local Truncation Absolue & Relative Error: %.12f %.12f\n',max(abs(abs_error_1)),max(abs(rel_error_1)));
end
```

- *Local Truncation error*
- $\theta_{i+2,2h} - \theta_{i,2h} = \frac{1}{6}2h(k_1 + 2k_2 + 2k_3 + k_4)$
- $\theta_{i+1,h} - \theta_{i,h} = \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4)$ -> $\theta_{i+2,h} - \theta_{i+1,h} = \frac{1}{6}h(k'_1 + 2k'_2 + 2k'_3 + k'_4)$
- Maximum absolute and relative error : $1x10^{-10}$ and $1x10^{-5}$

Accuracy

- Simple pendulum as test function
  - $\frac{d^2\theta}{dt^2} = -\frac{g}{l}\theta$
  - $\theta = \theta_{t=0} cos(\omega t) + \frac{\omega_{t=0}}{\omega_0} sin(\omega t)$
- True value calculated: -2.4941747244869 5 rad

ITERATIONS	STEP SIZE	EULER	ABS. ERROR	RK4	ABS. ERROR
101	0.1	**-4.090454359**	1.596279634	**-2.49417**	3.146287670E-06
201	0.05	**-3.19965216**	0.705477436	**-2.49417**	1.428671460E-07
401	0.025	**-2.825751259**	0.331576534	**-2.49417**	7.242349000E-09
801	1.25E-02	**-2.654927295**	0.160752571	**-2.49417**	3.998548159E-10
1601	6.25E-03	**-2.573323823**	0.079149099	**-2.49417**	2.334310523E-11
3201	3.13E-03	**-2.533446427**	0.039271702	**-2.49417**	1.413980044E-12
6401	1.56E-03	**-2.513735356**	0.019560632	**-2.49417**	8.437694987E-14

Error Analysis

$\varepsilon \propto h$

$\varepsilon \propto h^4$

STEP SIZE	ABS. ERROR FROM EULER'S	ERROR/2	ABS. ERROR FROM RK4	ERROR/2^4
0.1	1.596279634	0.798139817	3.146287670E-06	1.966429794E-07
0.05	0.705477436	0.352738718	1.428671460E-07	8.929196625E-09
0.025	0.331576534	0.165788267	7.242349000E-09	4.526468125E-10
0.0125	0.160752571	0.080376285	3.998548159E-10	2.499092600E-11
6.25E-03	0.079149099	0.039574549	2.334310523E-11	1.458944077E-12
3.13E-03	0.039271702	0.019635851	1.413980044E-12	8.837375276E-14
1.56E-03	0.019560632	0.009780316	8.437694987E-14	5.273559367E-15

Flow of the program

1. Variables, arrays initialization
2. Define set of 1^{st} order ODE to be solved
3. Apply selected ODE solver
    1. Check the local truncation error
    2. Compare results with MATLAB ode45 solver
4. Plot graphs
5. Output animation in gif format

Monitor my algorithm

- No analytical solution

1. $Local\ Truncation\ error = \theta_{i+2,2h} - \theta_{i+2,h}$
2. $Diviation\ from\ MATLAB\ ode45\ solver$

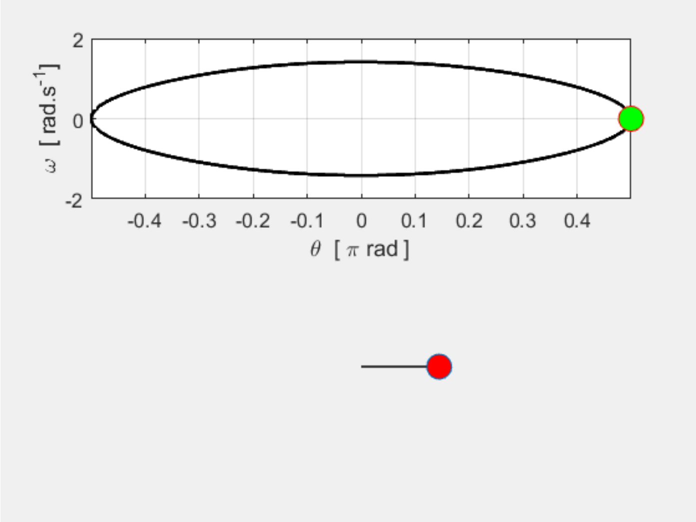
- Both absolute and relative errors

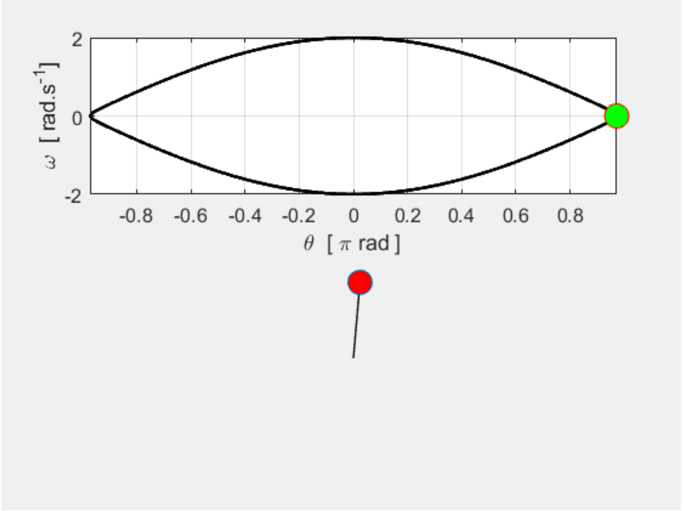
```
%=====================================================================
%Error with ode45
[t,y] = builtin_ode45();
abs_err = theta(end,1)-y(end,1);
rel_err =(theta(end,1)-y(end,1))/min(theta(end,1),y(end,1));
fprintf('Absolue & Relative Error With ode45: %.12f %.12f\n',abs_err,rel_err);
```

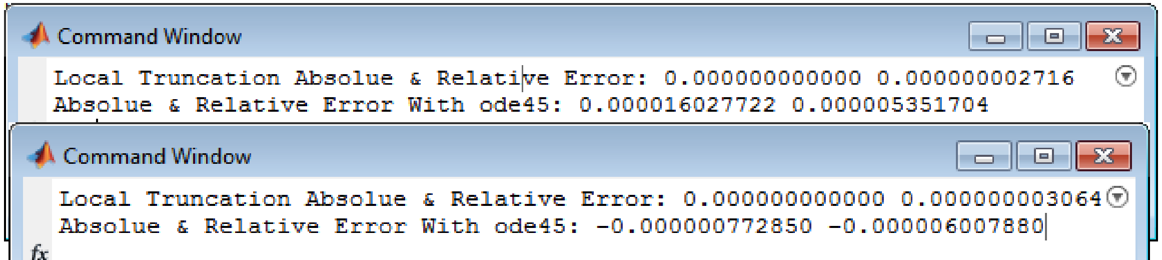
```
%=====================================================================
% MATLAB ode45
function [t,y] = builtin_ode45()
    options=odeset('RelTol',5.7627e-09,'AbsTol',1.1291e-10);
    [g,l,theta_0,w_0,~,~,~,dt,~,duration] = get_var();
    tspan= 0:dt:duration; % set time interval
    init=[theta_0,w_0]; % set initial conditions
    [t,y]=ode45(@myode,tspan,init,options);
    figure()
    plot(t,y(:,1)/pi);
end
function dydt = myode(t,y)
    [g,l,theta_0,w_0,Fext_0,Fext_w,c,~,~,~] = get_var();
    dydt = [y(2); -g/l.*sin(y(1))-c*y(2)+ Fext_0.*sin(Fext_w*(t))];
end
```

Pendulum without driving or dragging force

- Phase Space Plot (ω vs θ)
  - Orthogonal axes
  - Independent variables under study
- Trajectory cannot cross over itself
- Energy dynamics
- Attractors: Limit cycle, fixed point

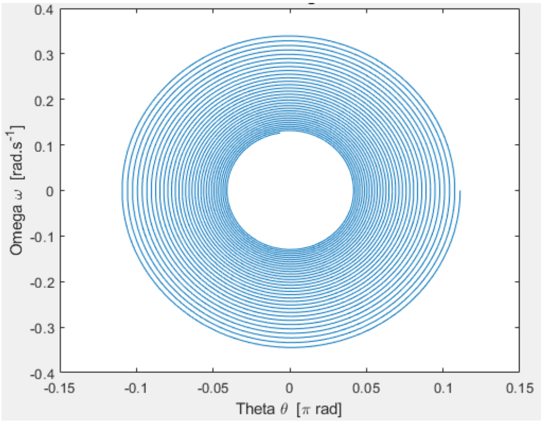




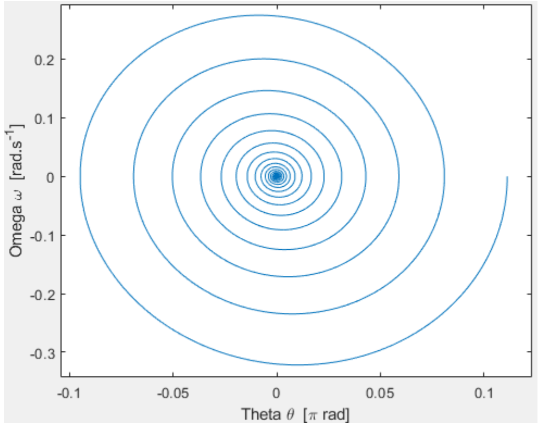


Pendulum without driving or dragging force

- Phase Space Plot (ω vs θ)
  - Orthogonal axes
  - Independent variables under study
- Trajectory cannot cross over itself
- Energy dynamics
- Attractors: Limit cycle, fixed point



$c = 0.01$



$c = 0.1$

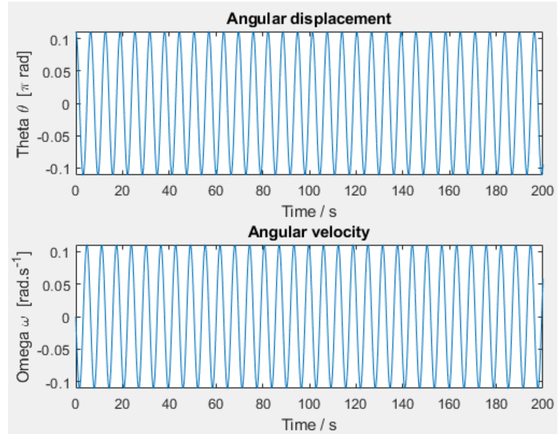
Driven Pendulum

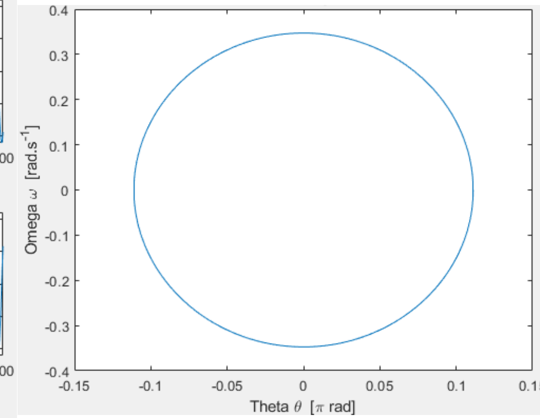
1. Limit Cycle
2. Period tripling
3. Chaos but bounded

- Route to chaos
- Aperiodic

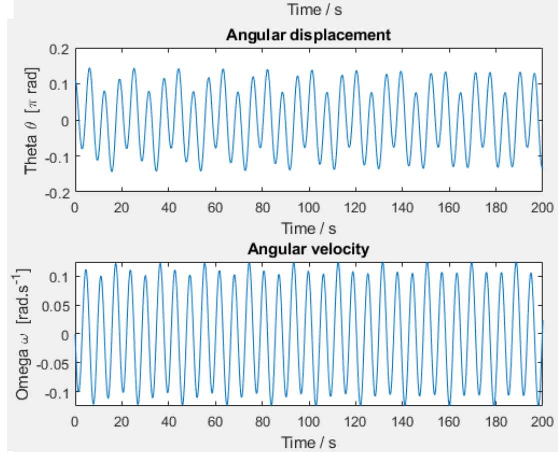
- Cross over each other
- Projection from 3D to 2D

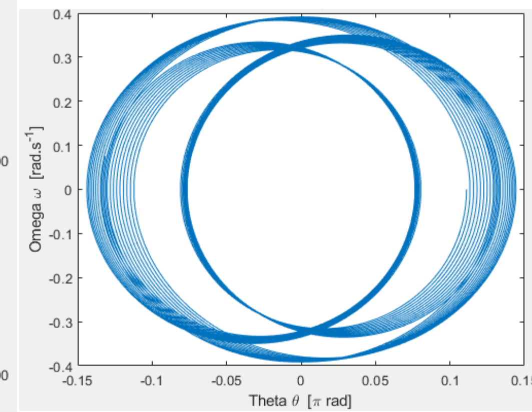
$F_{ext} = 0$



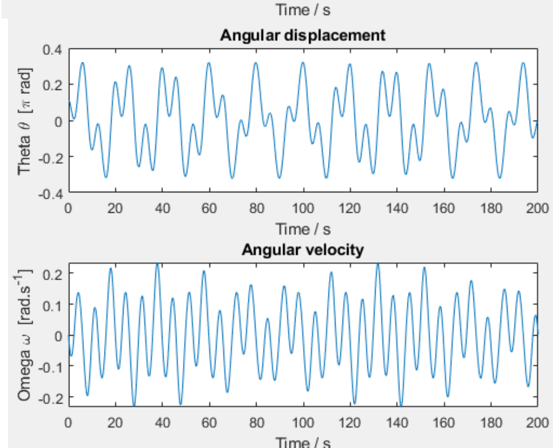


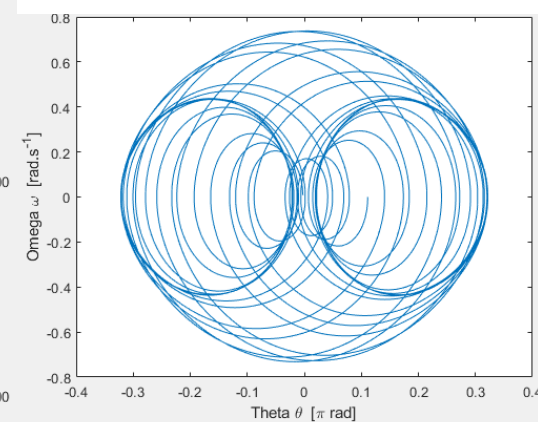
$F_{ext} = 0.1$





$F_{ext} = 0.5$



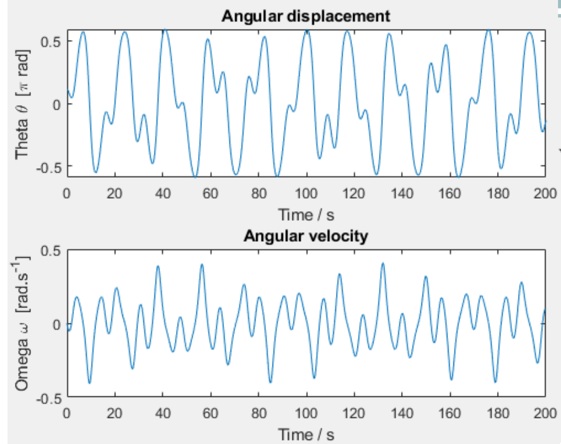


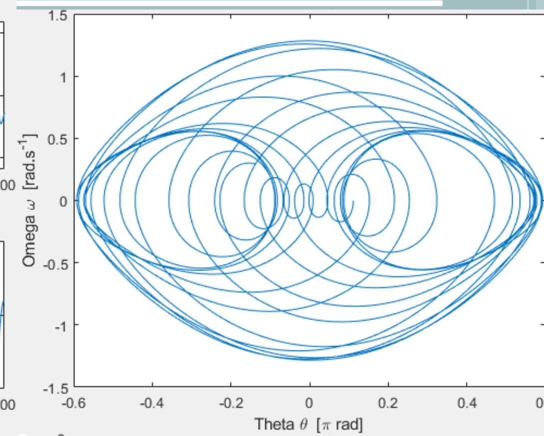
Driven Pendulum

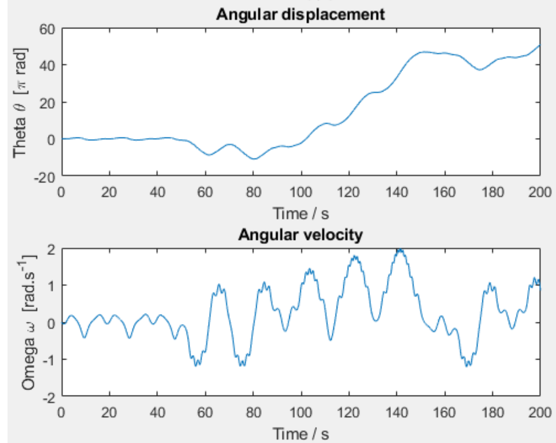
- Sensitive to initial conditions

1. Chaos but bounded $(-\pi, \pi)$
2. Complete rotation
3. Change of 0.0002 results in completely different trajectory

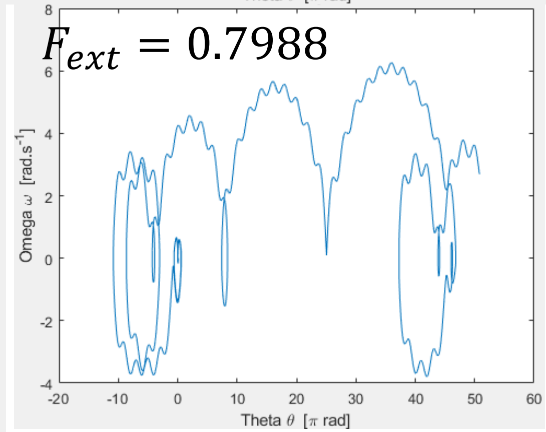
$F_{ext} = 0.79$

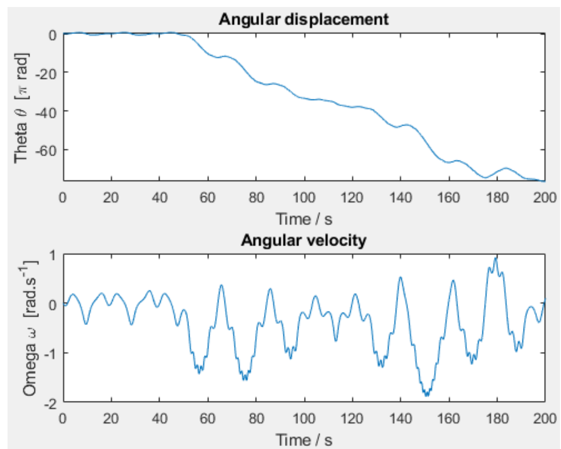


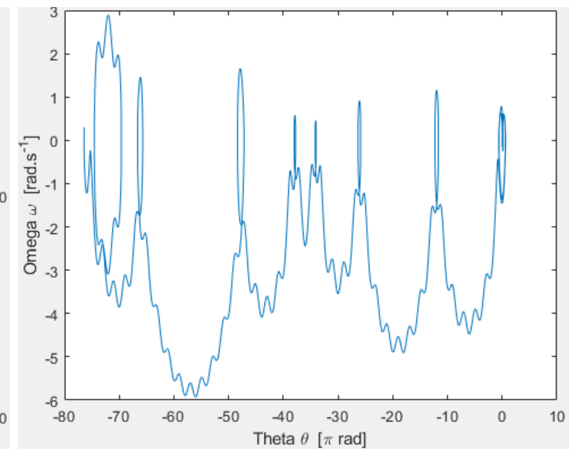




$F_{ext} = 0.7988$

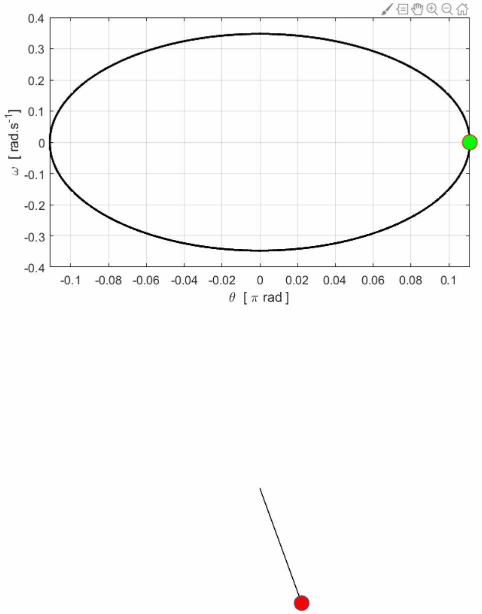




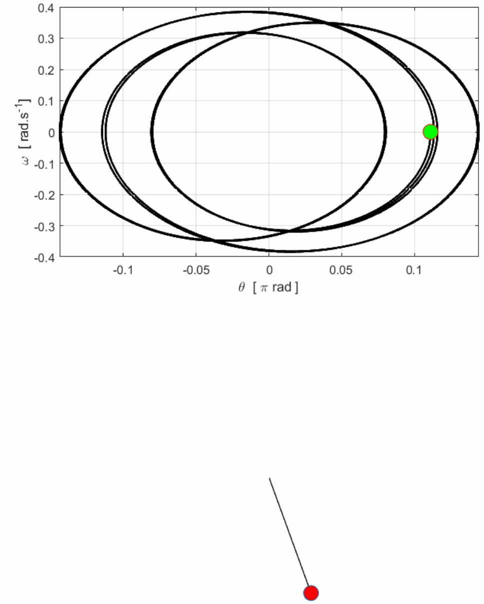


$F_{ext} = 0.799$

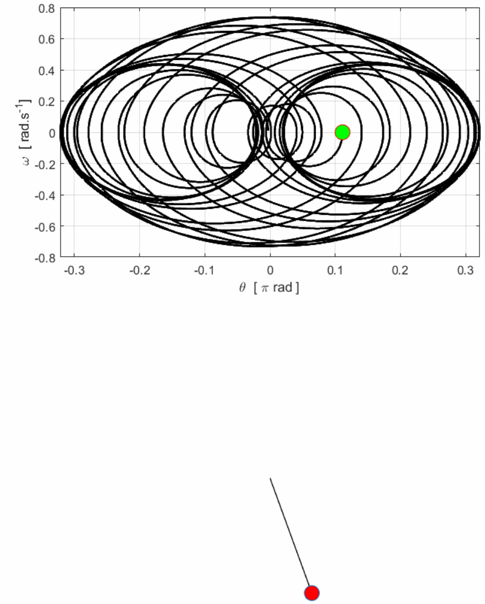
Driven Pendulum



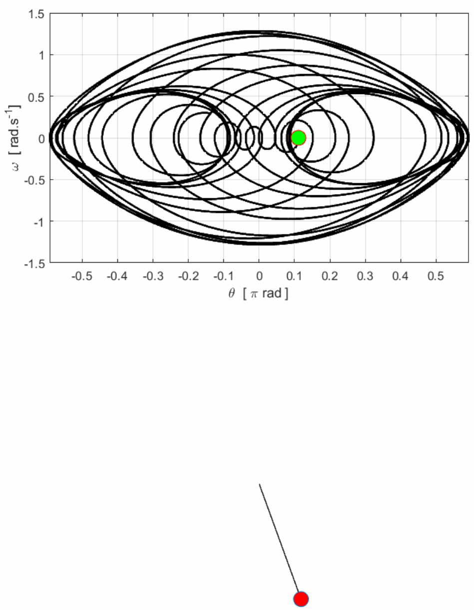
$F_{ext} = 0$



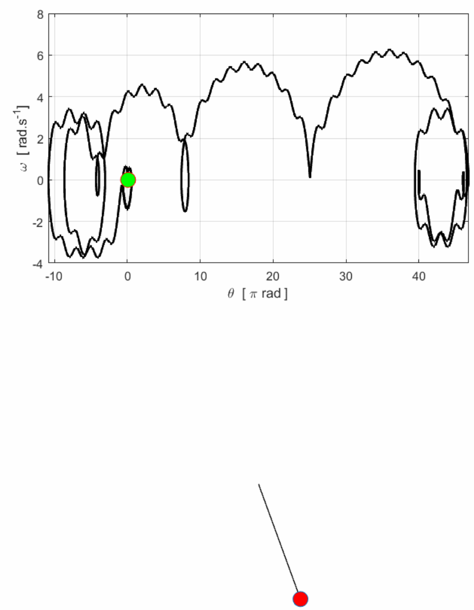
$F_{ext} = 0.1$



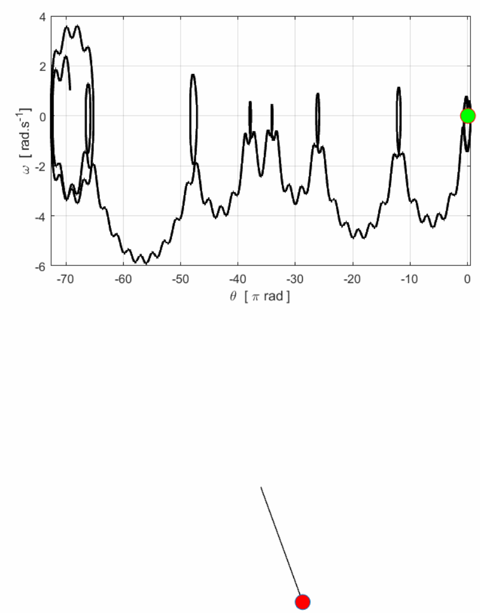
$F_{ext} = 0.5$



$F_{ext} = 0.79$



$F_{ext} = 0.7988$



$F_{ext} = 0.799$

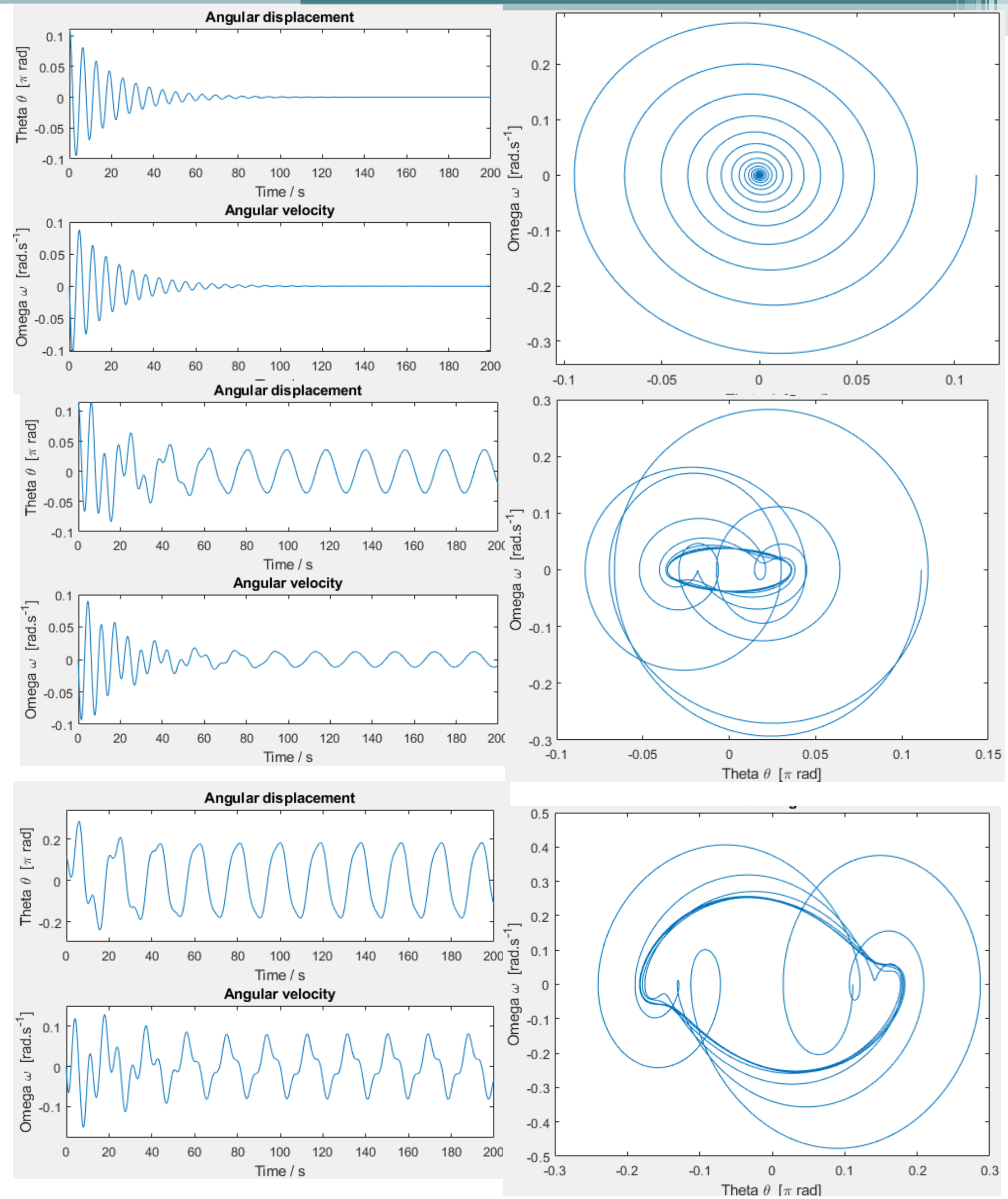
Driven damped Pendulum

$F_{ext} = 0$

- Chaotic behavior transition into attractors
- Bounded
  - Fixed Point
  - Limit cycle

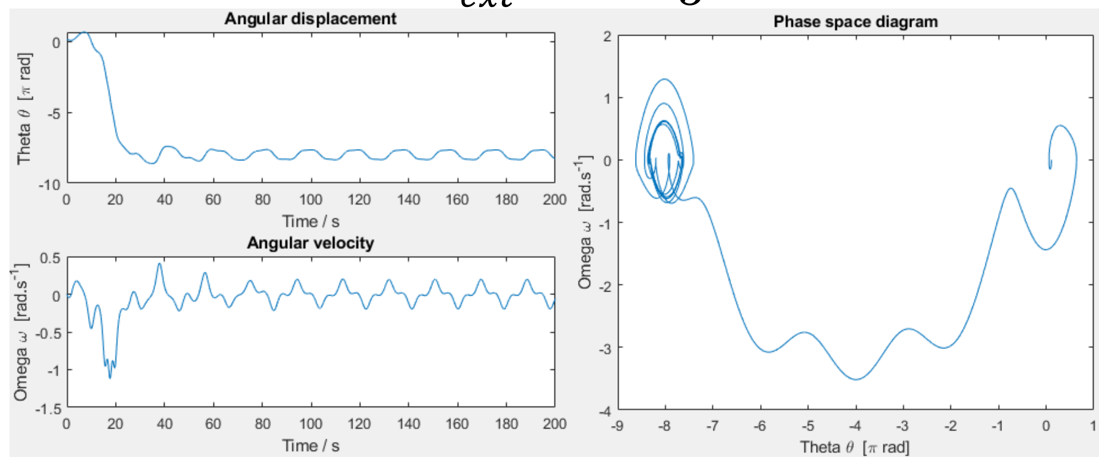
$F_{ext} = 0.1$

$F_{ext} = 0.5$

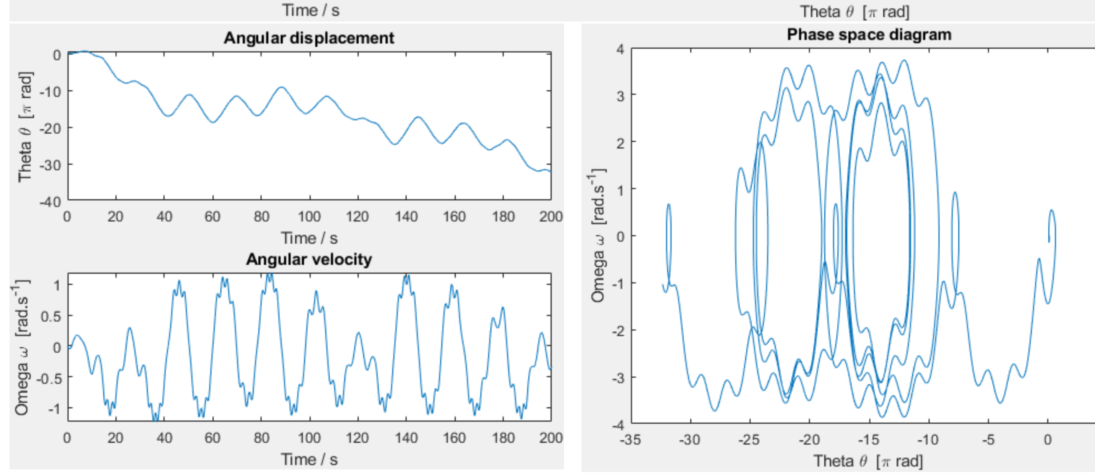


Driven damped pendulum

$F_{ext} = 0.885$

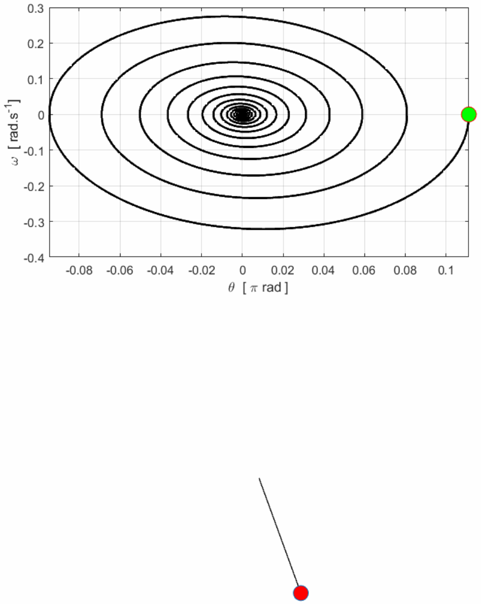


- Chaotic but not bounded transition into limit cycle
  - $(\theta, \omega) = (-8\pi, 0)$
- Slight variation of 0.001 shows completely different motion

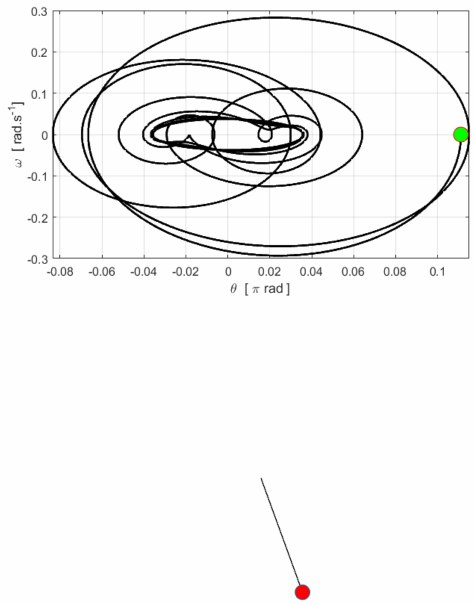


$F_{ext} = 0.886$

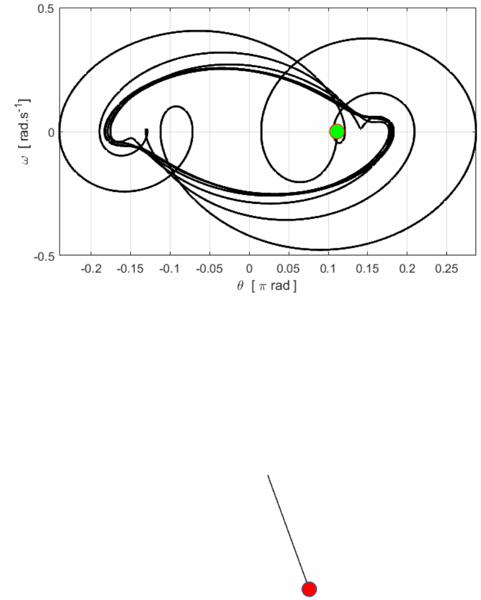
Driven damped pendulum



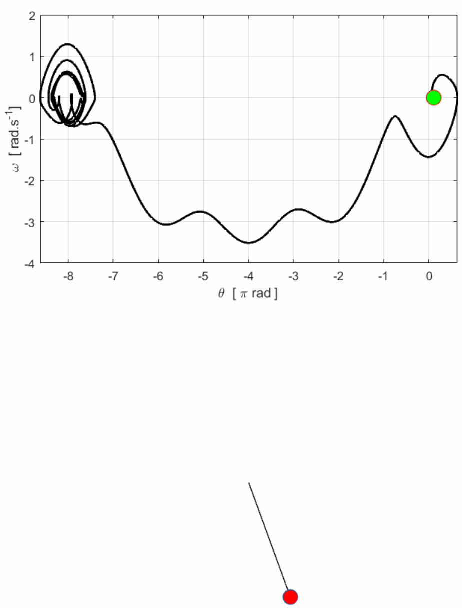
$F_{ext} = 0$



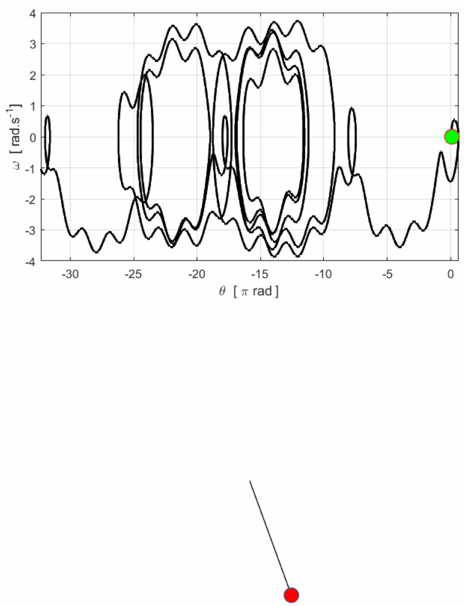
$F_{ext} = 0.1$



$F_{ext} = 0.5$



$F_{ext} = 0.885$

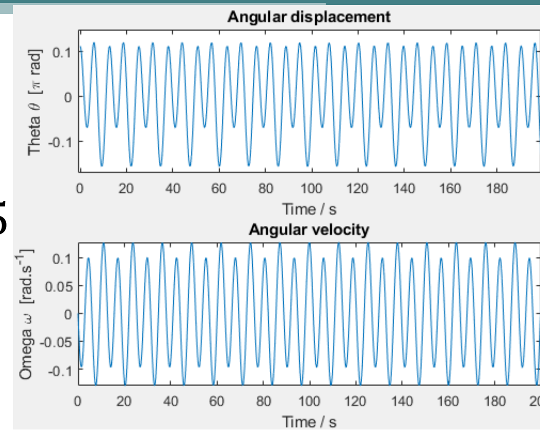


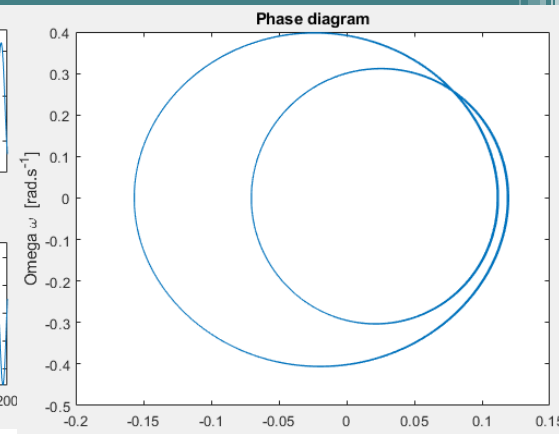
$F_{ext} = 0.886$

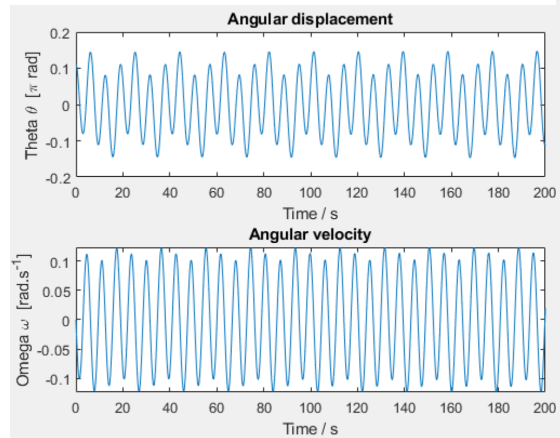
Period Doubling

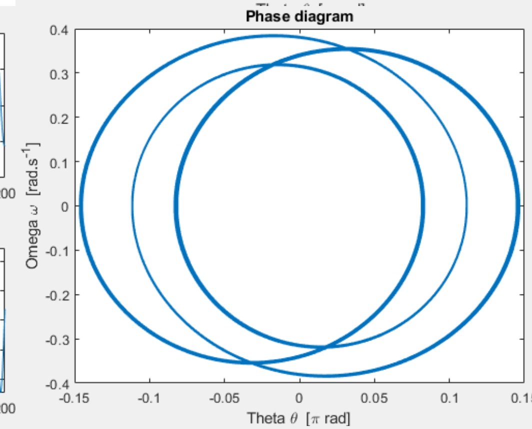
- Route to chaos
- Potential for chaotic behaviors
- Variation of ω_{ext}
- $\frac{2\pi}{n\omega_{\text{ext}}} = 6.347\text{s}$

$\omega_{ext} = 0.495$

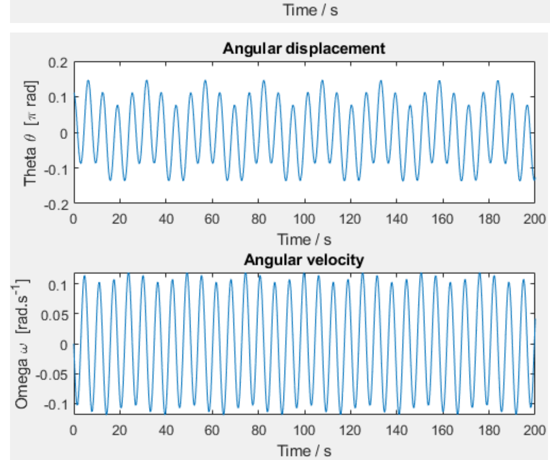


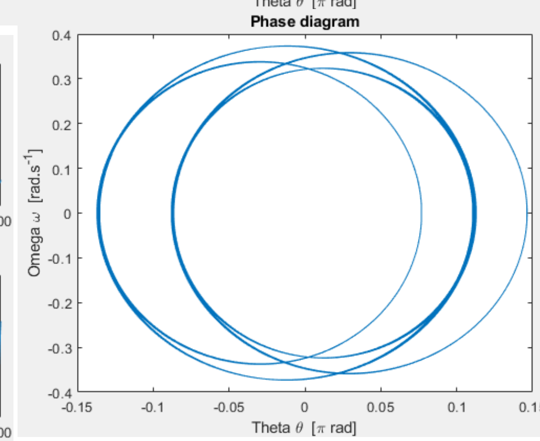






$\omega_{ext} = 0.33$

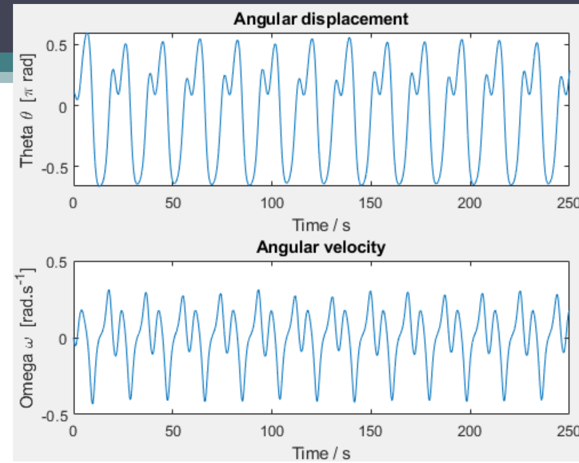


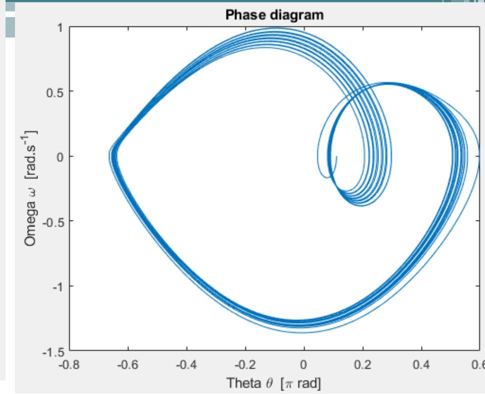


$\omega_{ext} = 0.248$

Period Doubling

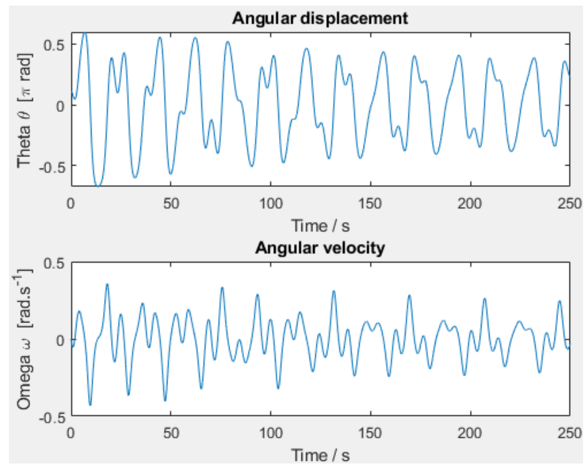
$$\omega_{ext} = \frac{1}{3}$$

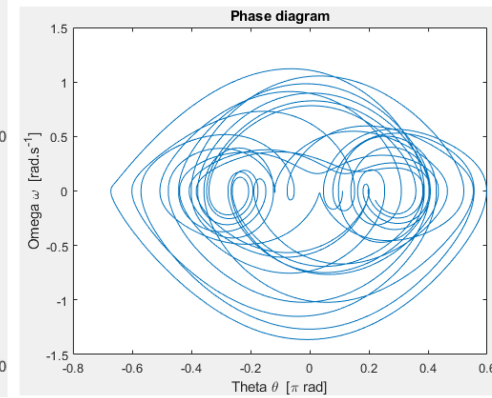




1. Period doubling
2. Chaos
3. Period quadrupling

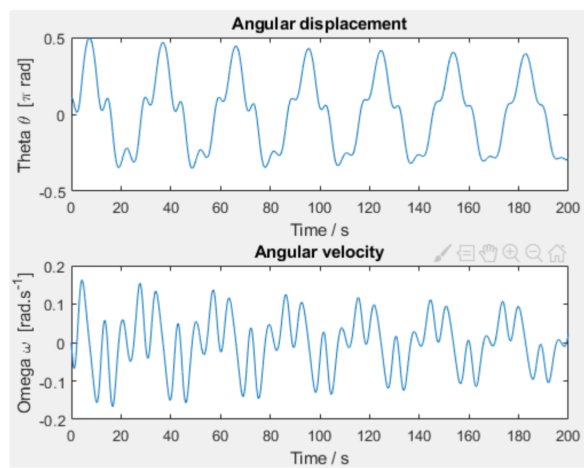
$$\omega_{ext} = 0.334$$

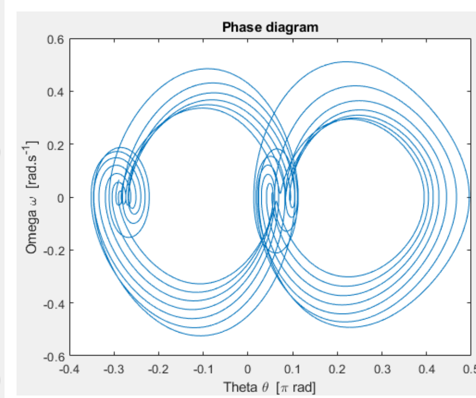




- Sensitive to initial conditions

$$\omega_{ext} = 0.215$$

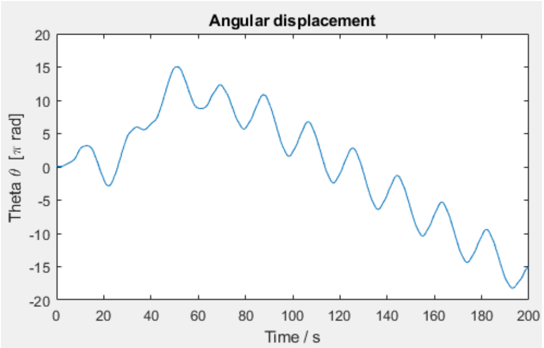


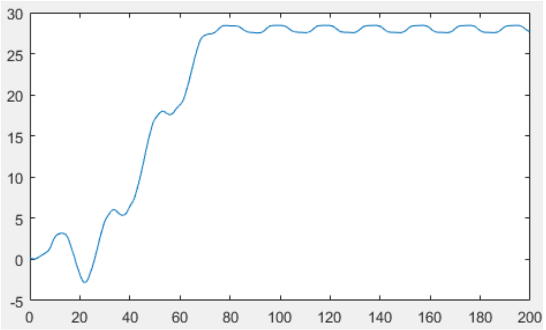


Error Analysis

- Comparison with Euler's and RK4 method
- Local Truncation Error
- Global Truncation Error compared with MATLAB ode45

RK4





ode45

$$\theta = 20°, \omega = 0, F_{ext} = 1, \omega_{ext} = \frac{1}{3}, c = 0.1,$$

$$step\ size = 0.003125 \text{ at } t = 0$$

Conclusion

- Simulation of the second order ODE
- Comparison of errors between Euler's and RK4 method
- Monitor errors arose between RK4 with higher order ODE solver MATLAB ode45
- Observe the path to chaos in driven pendulum
- Animation aids for better trajectory understanding

THANK YOU!