Chapter 19

Driven Damped Pendulum and Chaos

19.1 Equation of motion

The driven, damped pendulum is a relatively simple system that exhibits a wide range of interesting dynamical behaviors, including chaos.[1] There is no analytical solution for this system. It can only be investigated numerically or experimentally.

Figure 19.1 shows a plane pendulum with cord length ℓ and mass m. The angle between the pendulum cord and the vertical is Θ. The pendulum

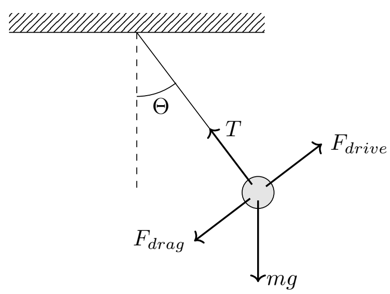

Fig. 19.1: A pendulum with mass m and cord length ℓ. The pendulum makes an angle Θ with respect to the vertical.

bob is subject to four forces: tension T from the cord, gravity mg, drag F_{drag} due to friction and/or air resistance, and a driving force F_{drive}. The motion is determined from Newton's second law.

[1] A readable account of the driven damped pendulum and chaos can be found in Taylor, J.R. (2005), *Classical Mechanics* (University Science Books).

Since the motion of the pendulum bob is always perpendicular to the cord, we only need to consider the perpendicular component of Newton's second law. The sum of forces perpendicular to the cord is

$$\sum F_\perp = F_{drive} + F_{drag} - mg\sin\Theta \tag{19.1}$$

where $mg\sin\Theta$ is the perpendicular component of the gravitational force. By Newton's second law, $\sum F_\perp$ equals the product of mass m and acceleration. Recall that the velocity of an object moving on a circlular path, like the pendulum bob, is $\ell\dot{\Theta}$ where the dot denotes a time derivative. Thus, the acceleration is $\ell\ddot{\Theta}$ and we have

$$m\ell\ddot{\Theta} = F_{drive} + F_{drag} - mg\sin\Theta \ . \tag{19.2}$$

This is a second order ordinary differential equation for Θ as a function of time t.

Let's assume the drag force is a linear function of the velocity $\ell\dot{\Theta}$, and that it opposes the motion. Thus,

$$F_{drag} = -\beta\dot{\Theta} \tag{19.3}$$

where β is some positive constant. Let the external driving force be a periodic function of time,

$$F_{drive} = \gamma\cos(2\pi t/\tau) \ , \tag{19.4}$$

where γ is a constant amplitude and τ (also a constant) is the time period of the force. Then the differential equation for the driven, dampled pendulum becomes

$$\ddot{\Theta} = -(g/\ell)\sin\Theta - \beta/(m\ell)\dot{\Theta} + \gamma/(m\ell)\cos(2\pi t/\tau) \ . \tag{19.5}$$

The degree of damping is controlled by β, the strength of the driving force is controlled by γ, and τ is the period of the driving force.

Exercise 19.1

Write the second order differential equation (19.5) as two first–order equations by introducing the new variable $\Omega = \dot{\Theta}$.

19.2 Long–term motion

Imagine pulling the pendulum aside to some initial angle $\Theta(0)$ and releasing it with some initial angular velocity $\Omega(0) = \dot{\Theta}(0)$. The pendulum will swing back and forth. If $\gamma = 0$, so the driving force is turned off, then damping will cause the pendulum to slow down and eventually come to rest at $\Theta = 0$. (Or equivalently, at some multiple of 2π.) This will happen for any initial conditions $\Theta(0)$ and $\Omega(0)$. If $\gamma \neq 0$, we would expect that in the long run, the pendulum's motion will be dictated by the driving force. That is, initially the pendulum's motion will depend on the particular values of $\Theta(0)$ and $\Omega(0)$, but at late times the pendulum's motion will be independent of initial conditions. It will move in some definite manner determined entirely by the driving force. The exact details of this long–term motion will depend on the parameters of the system.

Exercise 19.2

Use RK4 to solve for the motion of the driven, damped pendulum. Choose the parameter values (in SI units) $g = 9.8$, $m = 0.5$, $\ell = 0.11$, $\beta = 0.25$. Also let $\tau = 1.0$ and $\gamma = 1.0$. Carry out two simulations using very different initial conditions for $0 \leq t \leq 20$. On the same graph, plot Θ versus t for both simulations. What are the period and amplitude for t greater than (roughly) 2 or 3? What are the period and amplitude with $\tau = 1.0$ and $\gamma = 0.5$? With $\tau = 2.0$ and $\gamma = 1.0$?

Your results should confirm that after a few drive periods the motion of the pendulum settles into regular, nearly sinusoidal oscillations about $\Theta = 0$ (or some multiple of 2π) with period τ. This long–term solution is called an *attractor*. For the sets of parameters we have investigated thus far, there is unique attractor. The pendulum will aways evolve toward this attractor, independent of initial conditions.

19.3 Code tests

The driven damped pendulum and other chaotic systems can be very difficult to simulate. We need to make sure our results are reliable.

If we had an analytical solution for the system, we could carry out a convergence test: (1) compute the error at various resolutions N; (2) plot $\log|\text{error}|$ versus $\log(N)$; (3) find the slope. Since the errors for RK4 are proportional to N^{-4}, the slope should be approximately -4. This would confirm that the truncation errors are decreasing and the numerical results are becoming more and more precise as N is increased.

Unfortunately we don't have an analytical solution. But we can carry out a three–point convergence test as discussed in Sec. 17.5 in the context of Euler's method. Here's the idea: Let $\Theta_{(N)}$ denote the final value of Θ (the value at the final time) obtained from a simulation with N timesteps. Since the errors in RK4 are proportional to N^{-4}, we have

$$\Theta_{(N)} = \Theta_{(exact)} + C/(N)^4 \ , \tag{19.6a}$$

$$\Theta_{(2N)} = \Theta_{(exact)} + C/(2N)^4 \ , \tag{19.6b}$$

$$\Theta_{(4N)} = \Theta_{(exact)} + C/(4N)^4 \ , \tag{19.6c}$$

Combine these relations to obtain

$$\frac{\Theta_{(N)} - \Theta_{(2N)}}{\Theta_{(2N)} - \Theta_{(4N)}} = 2^4 \ . \tag{19.7}$$

This is an approximate result, because Eqs. (19.6) aren't exact; they omit terms proportional to higher powers of $1/N$. However, in the limit as N becomes large, the ratio on the left of Eq. (19.7) should approach the value $2^4 = 16$.

Exercise 19.3a

Continue with parameter values $g = 9.8$, $m = 0.5$, $\ell = 0.11$, $\beta = 0.25$, $\tau = 1.0$ and $\gamma = 1.0$ for the driven, damped pendulum. Use a time interval $0 \le t \le 10.0$ and any initial conditions. Your code will need to simulate the system at resolutions N, $2N$ and $4N$, then compute the ratio (19.7) using the final values of Θ. Is the ratio close to 16? Does it come closer to 16 as N is increased?

If the ratio approaches 16 for large N, then your results are converging and are probably reliable. Keep in mind, if N is *too* large, then machine roundoff errors can spoil the results.

Another way to check your code is to compare it with another code.

Exercise 19.3b

(Use $g = 9.8$, $m = 0.5$, $\ell = 0.11$, $\beta = 0.25$, $\tau = 1.0$ and $\gamma = 1.0$, and any initial conditions.) Simulate the driven, damped pendulum for times $0 \le t \le 10$ using your RK4 code and the built–in function `solve_ivp()` from the `scipy.integrate` library. Experiment with different solvers such as `RK45`, `DOP853`, *etc.* Plot the results from RK4 and `solve_ivp()` on the same graph. Can you see a difference? How large does N need to be for the plots to look the same?

If all has gone well, your RK4 code should agree with the various `solve_ivp()` solvers. By "agree" we mean that the curves on the graph look visually the same. This is not a definitive test of your code because we don't know how accurate the `solve_ivp()` solvers are. But it should give you some confidence that your code is performing correctly. After all, it is unlikely that your code and the `solve_ivp()` solvers are all wrong in the same way.

19.4 State space

For dynamical systems, the space of coordinates and velocities is called *state space*.[2] Each point in state space represents a unique state of the system. The initial data, in particular, correspond to a single point in state space. As the system evolves in time, its state changes and the initial data point sweeps out a curve through state space.

The periodicity of the motion for the driven, damped pendulum can been seen in a state space plot. Once the pendulum has settled into a nice periodic motion, a plot of Ω versus Θ should show a simple "orbit" that repeats once for each drive period. Figure 19.2 shows three orbits using various values of τ and γ.

You can display the state space orbit by using only the data from late times. For example, you might want to plot data for times that include only the last 4 drive cycles, $t_f - 4\tau \leq t \leq t_f$, where t_f is the final time. Define the "points per cycle" ppc $= N\tau/t_f$, which is the number of data points included in one drive period. (The initial time is 0.) Create a state space plot with

```
ppc = int(N*tau/tfinal)
plt.plot(Theta[N - 4*ppc:N+1],Omega[N - 4*ppc:N+1])
```

(This assumes `matplotlib.pyplot` has been loaded as `plt`.) The array `Theta[N-4*ppc:N+1]` consists of the final `4*ppc` elements of `Theta`, namely, `Theta[N-4*ppc]` through `Theta[N]`. To plot the final `n` drive cycles, use `Theta[N-n*ppc:N+1]`.

Exercise 19.4a

Create a state space plot for the driven, damped pendulum using only the data from late times. (Use $g = 9.8$, $m = 0.5$, $\ell = 0.11$,

[2] Equivalently, the space of coordinates and momenta is called *phase space*.

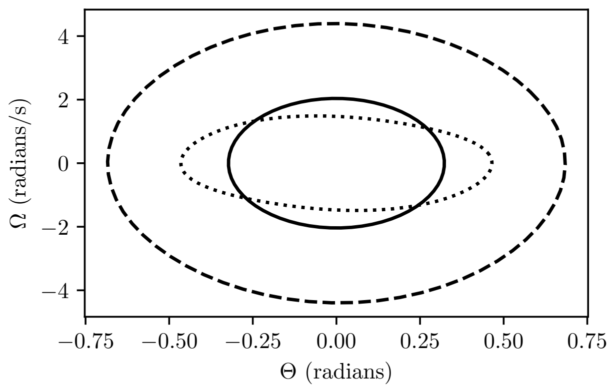

Fig. 19.2: State space orbits for the driven, damped pendulum with driving force parameters $\tau = 1$, $\gamma = 1$ (solid), $\tau = 1$, $\gamma = 2$ (dashed) and $\tau = 2$, $\gamma = 2$ (dotted).

$\beta = 0.25$, $\tau = 1.0$ and $\gamma = 1.0$.) Start with initial conditions $\Theta(0) = \Omega(0) = 0$. Choose the time interval $0 \le t \le 20$, and use enough timesteps to obtain a reliable result. The graph should show a simple closed orbit that repeats once each drive period.

As the driving force amplitude γ is increased, the pendulum exhibits a variety of interesting behaviors.

Exercise 19.4b

Continue using parameter values $g = 9.8$, $m = 0.5$, $\ell = 0.11$, $\beta = 0.25$ and $\tau = 1.0$, initial conditions $\Theta(0) = \Omega(0) = 0$, and time interval $0 \le t \le 20$. Let $\gamma = 5.15$. Create a state space plot as well as a plot of Θ versus t. Describe the motion qualitatively. What is the period of the motion?

19.5 Period doubling

Set the initial conditions to $\Theta(0) = -\pi/2$, $\Omega(0) = 0$. Continue to use parameter values $g = 9.8$, $m = 0.5$, $\ell = 0.11$, $\beta = 0.25$ and $\tau = 1.0$.

Exercise 19.5a

Let $\gamma = 5.0$. Create a phase space plot and a plot of Θ versus t. Describe the motion qualitatively. What is the period of the motion?

As the drive amplitude γ is increased beyond about 5.0568, the pendulum exhibits *period doubling*. The pendulum executes a more complicated dynamical behavior with period 2τ, twice the drive period. With further increase in γ the period doubles again, to 4τ. Then 8τ, 16τ, *etc.* The sequence of period doubling ends at $\gamma \approx 5.1341$. Beyond this treshold, the pendulum motion is not periodic at all—it is chaotic. The range of dynamical behaviors is depicted in Fig. 19.3.

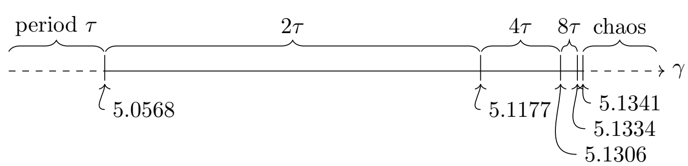

Fig. 19.3: Period doubling. As the drive amplitude γ is increased, the period doubles from τ to 2τ, then 4τ, *etc.* Motion with period 16τ, 32τ, *etc.* is squeezed into the narrow range $5.1334 \lesssim \gamma \lesssim 5.1341$. Beyond $\gamma \approx 5.1341$, the motion is chaotic. The threshold values of γ are approximate.

Exercise 19.5b

Choose a value of γ in the middle of the range $5.0568 \lesssim \gamma \lesssim 5.1177$, where the motion of the pendulum has period is 2τ. Be careful that your resolution is sufficiently high to obtain accurate results. You might want to extend your simulation to a final time of $t_{final} = 100.0$ or more, to make sure you're seeing the long–term behavior. Use state space and Θ versus t plots to interpret the results. If your Python application allows for interactive graphs, it can be helpful to zoom in on small sections of these plots.

Exercise 19.5c

Choose a value of γ in the middle of the range $5.1177 \lesssim \gamma \lesssim 5.1306$, where the motion of the pendulum has period 4τ. Examine the motion using state space and Θ versus t plots.

These simulations are difficult to carry out for several reasons. As always, simulations must be run at more than one resolution to help insure that truncation error is not affecting the results. If N is too large, machine roundoff errors can be a problem. An additional difficulty is that the ideal steady–state motion (the attractor solution) is never actually reached with a finite run time. As we progress to more complex motion, with period 2τ, 4τ, *etc.*, the periodic behavior takes longer to emerge and is more difficult to identify in a finite run time. Of course we can increase the run time, but then the truncation errors become larger.

The threshold values shown in Fig. 19.3 are approximate. They were obtained from an RK4 code with run time $0 \le t \le 400$, at resolution $N = 400000$ and $N = 800000$. To identify the onset of the first period doubling, the values of Θ at late integer times $360, 361, \ldots, 400$ were saved. The code then produces a plot of $\Theta - \Theta_{ave}$, where Θ are the saved values and Θ_{ave} is the average of these values. If the motion had exactly period $\tau = 1$, the plot would show a horizontal straight line. If the motion had exactly period $2\tau = 2$, the motion would be a zig–zag between two distinct values. Unfortunately, the threshold between period $\tau = 1$ and period $2\tau = 2$ is not distinct, because the motion is not precisely periodic (due to the finite run time). What we actually see is, for $\gamma \lesssim 5.0568$, a zig–zag pattern with decreasing amplitude. If we could run the code for a much longer time, we would expect the amplitude to decrease to zero, indicating a period $\tau = 1$ motion. For $\gamma \gtrsim 5.0568$ the zig–zag pattern is steady, with a constant amplitude. This indicates a true period $2\tau = 2$ motion.

The threshold values between higher period motions were obtained using the same strategy. For example, for the threshold between 2τ and 4τ, late time values of Θ were saved at times separated by period 2, namely, $360, 362, \ldots, 400$. The difference between the saved values and their average was plotted against time. A decreasing amplitude zig–zag pattern indicates motion with period $2\tau = 2$. A constant amplitude zig–zag pattern indicates motion with period $4\tau = 4$.

Exercise 19.5d

Use your RK4 code to find the threshold value of γ between period τ and period 2τ. You can use the strategy above. Can you think of a better way to determine the threshold? Do you agree with the result $\gamma = 5.0568$?

You can see from Fig. 19.3 that the threshold values of γ bunch together as the period increases. Table 19.1 shows the range of values of γ for motion with period 2τ, 4τ, and 8τ. (The transition from 8τ to 16τ occurs at $\gamma \approx 5.1334$; this is not shown in Fig. 19.3.) The ratio of successive values, from 2τ to 4τ, is $0.0609/0.0129 = 4.72$. The ratio of successive values from 4τ to 8τ is approximately the same, $0.0129/0.0028 = 4.61$. Careful experiments show that these ratios tend to a fixed value of approximately 4.669. This is called the *Feigenbaum constant*.

period	range of γ values
2τ	$5.1177 - 5.0568 = 0.0609$
4τ	$5.1306 - 5.1177 = 0.0129$
8τ	$5.1334 - 5.1306 = 0.0028$

Table 19.1: The ranges of values for the drive force amplitude γ that yield motion with periods 2τ, 4τ, and 8τ.

What is remarkable about the Feigenbaum constant is that it is *universal*. For a wide class of chaotic systems that pass through a sequence of period doublings on their way to chaos, the ratio of successive ranges of parameter values is given by the Feigenbaum constant.

19.6 Chaos

Keep the same initial data and parameter values as in the previous section.

For values of γ greater than about 5.1341, the driven, damped pendulum exhibits chaos. The late–time motion is chaotic rather than periodic.

Exercise 19.6a

Simulate the driven, damped pendulum over the time interval $0 \leq t \leq 20$ using RK4. Set the drive amplitude to $\gamma = 5.7$. Describe the motion qualitatively.

Simulating a chaotic system can be tricky. We need to make sure these results are reliable.

Exercise 19.6b

Carry out a three–point convergence test using the RK4 code from the previous exercise. Use the resolutions $N = 25000$, 50000 and 100000 and compute the ratio

$$\frac{\Theta_{(100000)} - \Theta_{(50000)}}{\Theta_{(50000)} - \Theta_{(25000)}} ,$$

where each Θ value is the angle of the pendulum at the final time $t = 20.0$. Is the ratio close to $2^4 = 16$?

You can compare your RK4 results with the built–in ODE solvers from the `scypy.integrate` library. These solvers are accessed with the function `solve_ivp()`, as discussed in Secs. 18.6 and 18.7. The accuracy of the `solve_ivp()` solvers is controlled by two parameters, the absolute tolerance `atol` and the relative tolerance `rtol`. The solvers will adjust the timestep in an attempt to keep the truncation error below `atol + rtol*abs(y)`, where `abs(y)` is the larger of the dependent variables Θ and Ω. The default values for the tolerances are `atol = 10**(-3)` and `rtol = 10**(-6)`. You can change these values to, say, 10^{-8} and 10^{-10} by adding the options `atol = 10**(-8), rtol = 10**(-10)` to the function `solve_ivp()`.

Exercise 19.6c

Simulate the pendulum with RK4 as before. Extend your code to include a simulation using one of the built–in `solve_ivp()` solvers. Plot the results Θ versus t from RK4 and from `solve_ivp()` on the same graph. Do the curves overlap? If not, try adjusting the absolute and relative tolerances in the `solve_ivp()` function. You can also try a different ODE solver.

19.7 Sensitivity to initial conditions

Chaotic systems can be extremely sensitive to initial conditions. You can monitor the sensitivity to initial conditions by creating a version of your code that runs two simulations, one with initial conditions $\Theta(0) = -\pi/2$, $\Omega(0) = 0$ and the other with initial conditions $\Theta(0) = -\pi/2 + \epsilon$, $\Omega(0) = 0$. Compute the difference in angles $\Delta\Theta$ between the two simulations, then plot $\ln|\Delta\Theta(t)|$ versus t.

First, let's look at an example with non–chaotic motion. With $\gamma \lesssim 5.1341$ and $\epsilon = 0.1$, a plot of $\ln|\Delta\Theta(t)|$ versus t will look similar to Fig. 19.4.

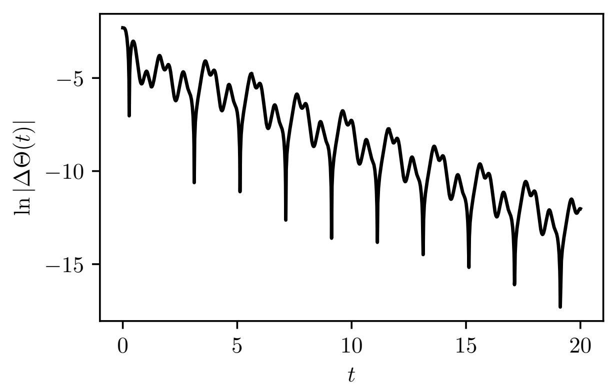

Fig. 19.4: A natural log plot showing the difference in angle for two simulations with different intial conditions. The downward trend shows that the differences decay away exponentially.

Note the downward pointing spikes. These occur when $\Delta\Theta$ changes sign and comes close to zero. If $\Delta\Theta$ happens to be exactly zero at some particular time, Python will complain that it can't evaluate $\ln|\Delta\Theta|$. This is an unlikely occurance. Most often, because the evolution takes place in discrete time steps, $\Delta\Theta$ will come close to zero as it changes sign but it won't exactly equal zero. The graph produces a sharp downward spike as $\Delta\Theta$ passes close to zero.

We're not actually interested in the spikes. What matters is the upper "envelope" of the curve. This shows a downward trend, decreasing from $\ln(0.1) \approx -2$ to about -12 in the time interval from 0 to 20. The envelope has a slope of roughly $(-12-(-2))/20 = -0.5$. This tells us that the angle difference decreases from its initial value of 0.1 as

$$\Delta\Theta(t) \approx 0.1 \cdot e^{-0.5t} \ . \tag{19.8}$$

The difference between angles decays away exponentially rapidly.

The slope -0.5 is called the *Lyapunov exponent*. When a Lyapunov exponent is negative, it characterizes the rate at which solutions with different initial conditions evolve toward a common attractor.

Exercise 19.7a

Confirm the results described above. Keep the same parameter values as in the previous sections, and choose a value of γ well below the chaotic regime (say, $\gamma \lesssim 5.1$). What value do you get for the Lyapunov exponent?

When the motion is chaotic, the angle difference increases in time and the Lyapunov exponent is positive. Solutions with different initial conditions rapidly diverge away from one another. The system is sensitive to initial conditions.

Exercise 19.7b

Let $\gamma = 5.7$ so the motion of the pendulum is chaotic. For the second set of initial conditions, choose $\epsilon = 10^{-10}$. Plot $\ln|\Delta\Theta(t)|$ versus t over the time interval $0 \leq t \leq 20$ and find the Lyapunov exponent.

While the angle difference is small, the two solutions will appear nearly identical. However, with the parameters set for chaotic motion, the angle difference evantually approaches a value of order unity and the solutions diverge.

Exercise 19.7c

Keep the parameter values as in the previous exercise, but set the time interval to $0 \leq t \leq 25$. Compare the evolution for the two sets of initial conditions by examining a phase space plot and a plot of Θ versus t. Describe the results.

The motion of a chaotic system like the driven, damped pendulum is deterministic. That is, for any initial point in state space, the differential equations define a unique phase space trajectory. Nevertheless, the motion of a chaotic system is unpredictable in the following sense: We would need to know the initial conditions to arbitrarily high accuracy to predict the exact motion for any extended time.

Why are chaotic systems so difficult to simulate numerically? The difficulty is that any numerical simulation is subject to errors. These errors depend on the numerical algorithm, the resolution, and the limits of machine precision. Any error, whether it occurs in the initial data or at a later time, changes the state of the system. An error at time t "bumps" the system point in phase space from one trajectory to another. If the two trajectories diverge exponentially, this error eventually grows to order unity and drastically alters the results.

One source of error is truncation error, which depends on the numerical algorithm and the resolution.

Exercise 19.7d

Keep the parameter values set to chaotic motion, as in the previous exercise, and initial conditions $\Theta(0) = -\pi/2$, $\Omega(0) = 0$. Set the final time to 100 or more. Compare results using your RK4 code at different resolution values N. Also compare with various `solve_ivp()` solvers, and different settings for the relative and absolute tolerances. Do you trust the results from any of these simulations?

You can reduce the truncation error by increasing the resolution. However, there is a limit to how much the errors can be reduced, due to machine roundoff errors.

Exercise 19.7e

Modify your RK4 code from the previous exercise to run two or more simulations "side–by–side." Compare different ways of writing the equations of motion. For example, you could write the second term in Eq. (19.5) as either `beta/(m*l)*Omega` or `beta*Omega/(m*l)`. Or change the order of the three terms in Eq. (19.5). Why should these changes affect the results? Can you make the differences dissappear by increasing the resolution?