# Lecture 29
# Reduction of Higher Order Equations to Systems

### The motion of a pendulum

Consider the motion of an ideal pendulum that consists of a mass $m$ attached to an arm of length $\ell$. If we ignore friction, then Newton's laws of motion tell us

$$m\ddot{\theta} = -\frac{mg}{\ell}\sin\theta,$$

where $\theta$ is the angle of displacement.

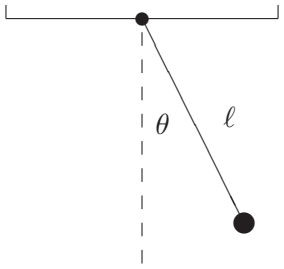


Figure 29.1: A pendulum.

If we also incorporate moving friction and sinusoidal forcing then the equation takes the form

$$m\ddot{\theta} + \gamma\dot{\theta} + \frac{mg}{\ell}\sin\theta = A\sin\Omega t.$$

Here $\gamma$ is the coefficient of friction and $A$ and $\Omega$ are the amplitude and frequency of the forcing. Usually, this equation would be rewritten by dividing through by $m$ to produce

$$\ddot{\theta} + c\dot{\theta} + \omega\sin\theta = a\sin\Omega t, \tag{29.1}$$

where $c = \gamma/m$. $\omega = g/\ell$ and $a = A/m$.

This is a second order ODE because the second derivative with respect to time $t$ is the highest derivative. It is nonlinear because it has the term $\sin\theta$ and which is a nonlinear function of the dependent variable $\theta$. A solution of the equation would be a function $\theta(t)$. To get a specific solution we need side conditions. Because it is second order, 2 conditions are needed, and the usual conditions are initial conditions

$$\theta(0) = \theta_0 \qquad \text{and} \qquad \dot{\theta}(0) = v_0. \tag{29.2}$$

## Converting a general higher order equation

All of the standard methods for solving ordinary differential equations are intended for first order equations. For this reason, it is inconvenient to solve higher order equations numerically. However, most higher-order differential equations that occur in applications can be converted to a *system* of first order equations and that is what is usually done in practice.

Suppose that an $n$-th order equation can be solved for the $n$-th derivative, i.e. it can be written in the form

$$x^{(n)} = f\left(t, x, \dot{x}, \ddot{x}, \ldots, \frac{d^{n-1}x}{dt^{n-1}}\right).$$

Then it can be converted to a first-order system by this standard change of variables:

$$\begin{aligned} y_1 &= x \\ y_2 &= \dot{x} \\ &\vdots \\ y_n &= x^{(n-1)} = \frac{d^{n-1}x}{dt^{n-1}}. \end{aligned}$$

The resulting first-order system is

$$\begin{aligned} \dot{y}_1 &= \dot{x} = y_2 \\ \dot{y}_2 &= \ddot{x} = y_3 \\ &\vdots \\ \dot{y}_n &= x^{(n)} = f(t, y_1, y_2, \ldots, y_n). \end{aligned}$$

In vector form this is simply $\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y})$ with $f_i(t, \mathbf{y}) = y_{i+1}$ for $i < n$ and $f_n(t, \mathbf{y}) = f(t, y_1, y_2, \ldots, y_n)$.

For the example of the pendulum (29.1) the change of variables has the form

$$\begin{aligned} y_1 &= \theta \\ y_2 &= \dot{\theta}, \end{aligned}$$

and the resulting equations are

$$\begin{aligned} \dot{y}_1 &= y_2 \\ \dot{y}_2 &= -cy_2 - \omega \sin(y_1) + a \sin(\Omega t). \end{aligned} \tag{29.3}$$

In vector form this is

$$\dot{\mathbf{y}} = \begin{pmatrix} y_2 \\ -cy_2 - \omega \sin(y_1) + a \sin(\Omega t) \end{pmatrix}.$$

The initial conditions are converted to

$$\mathbf{y}(0) = \begin{pmatrix} y_1(0) \\ y_2(0) \end{pmatrix} = \begin{pmatrix} \theta_0 \\ v_0 \end{pmatrix}. \tag{29.4}$$

As stated above, the main reason we wish to change a higher order equation into a system of equations is that this form is convenient for solving the equation numerically. Most general software for solving ODEs (including MATLAB) requires that the ODE be input in the form of a first-order system. In addition, there is a conceptual reason to make the change. In a system described by a higher order equation, knowing the position is not enough to know what the system is doing. In the case of a second order equation, such as the pendulum, one must know both the angle and the angular velocity to know what the pendulum is really doing. We call the pair $(\theta, \dot{\theta})$ the *state* of the system. Generally in applications the vector $\mathbf{y}$ is the state of the system described by the differential equation.

## Using Matlab to solve a system of ODE's

In MATLAB there are several commands that can be used to solve an initial value problem for a system of differential equations. Each of these correspond to different solving methods. The standard one to use is `ode45`, which uses the algorithm "Runge-Kutta 4 5". We will learn about this algorithm later.

To implement `ode45` for a system, we have to input the vector function $f$ that defines the system. For the pendulum system (29.3), we will assume that all the constants are 1, except $c = .1$, then we can input the right hand side as

```
> f = inline('[y(2);-.1*y(2)-sin(y(1))+sin(t)]','t','y')
```

The command `ode45` is then used as follows:

```
> [T Y] = ode45(f,[0 20],[1;-1.5]);
```

Here `[0 20]` is the time span you want to consider and `[1;-1.5]` is the initial value of the vector `y`. The output `T` contains times and `Y` contains values of the vector `y` at those times. Try

```
> size(T)
> T(1:10)
> size(Y)
> Y(1:10,:)
```

Since the first coordinate of the vector is the position (angle), we are mainly interested in its values:

```
> theta = Y(:,1);
> plot(T,theta)
```

In the next two sections we will learn enough about numerical methods for initial value problems to understand roughly how MATLAB produces this approximate solution.

## Exercises

29.1 Consider the pendulum system but with no friction or forcing, i.e. $\gamma = A = 0$. What would equation (29.3) become in this case? Use the last example to solve the system with the initial condition $[\theta_0, 0]'$ for $\theta_0 = .1\pi$. Use the plot of the solution to find the frequency of the pendulum with this initial condition. Do the same for $\theta_0 = .5\pi$ and $.9\pi$. How does the frequency depend on the amplitude of a pendulum?

29.2 Transform the ODE

$$\dddot{x} + \ddot{x}^2 - 3\dot{x}^3 + \cos^2 x = e^{-t}\sin(3t)$$

into a first order system. Suppose the initial conditions for the ODE are $x(1) = 1$, $\dot{x}(1) = 2$, and $\ddot{x}(1) = 0$. Find a numerical solution of this IVP using `ode45` and plot the first coordinate ($x$). Try time intervals `[1 2]` and `[1 2.1]` and explain what you observe. (Remember to use entry-wise operations, `.*` and `.^`, in the definition of the vector function.)