Johnathan von der Heyde - 2019

Project 3 – Simple Pendulum Dynamics

Pendulums are prototypical physical systems. Normally they behave well and are thus excellent pedagogical tools for introductory Physics courses. Upon closer investigation however, they prove to be remarkably complex. Nonetheless, the investigation of the pendulum remains a highly educational exercise – especially for those interested in the surprising subtleties this seemingly simple system offers.

Here we investigate some of those subtleties, with emphasis on the Numerical Methods that make it all possible.

Starting as usual with Newton's Law,

$$F = ma$$

We consider the force felt by the pendulum's mass, and realize any radial force is counter balanced by tension in the supporting string,

$$F_r = T$$

In 2D, that only leaves us to consider azimuthal forces, i.e., F_θ. And with mild investigation we recognize this as none other than

$$F_\theta = -mgsin\theta$$

That is, the radial component of the gravitational force, with a negative sign since it is a *restoring force*, seeking its own minimization. Simplifying,

$$ma = -mgsin\theta$$
$$a = -gsin\theta$$

Importantly, the acceleration here needs to be understood as $\frac{d^2s}{dt^2}$, where $s = \ell\theta$, the arch length, with ℓ being the length of the string. Therefore, we have our 2nd order differential in terms of θ

$$\frac{d^2\theta}{dt^2} = -\frac{g}{\ell}sin\theta$$

Recognizing angular frequency, $\omega = \frac{d\theta}{dt}$, we have what we need

$$\Rightarrow d\theta = \omega * dt$$
$$\Rightarrow d\omega = -\frac{g}{\ell}sin\theta * dt$$

These two 1st order differential equations can now be solved numerically.

However, one is distinctively non-linear, so in order to make this pendulum behave well, we impose the *small angle approximation*: $sin\theta \approx \theta$. This leaves us with a simple situation.

$$d\theta = \omega * dt$$
$$d\omega = -\frac{g}{\ell}\theta * dt$$

Numerically, $dx = x_{i+1} - x_i$ or equivalently $dx = x_i - x_{i-1}$ therefore,

$$\theta_i = \theta_{i-1} + \omega_{i-1} * \Delta t$$

$$\omega_i = \omega_{i-1} - \frac{g}{\ell}\theta_{i-1} * \Delta t$$

Where Δt is our time-step, i.e., the resolution of our calculation $\Delta t = t_i - t_{i-1} \forall\ t$

This is *Euler's Method* and it yields false results, as we'll see. Cromer found the issue and solved it. Turns out we need an *order of operations*.

Meaning, whereas current ω can rely on previous θ, current θ *cannot* rely on previous ω, and must instead rely on current ω. This enforcing of a particular order of operations grants our system a temporally unambiguous quality, i.e., it is now *reversible*, which means it necessarily conserves energy.

Therefore the *Cromer-Euler Method* becomes

$$\omega_i = \omega_{i-1} - \frac{g}{\ell}\theta_{i-1} * \Delta t$$

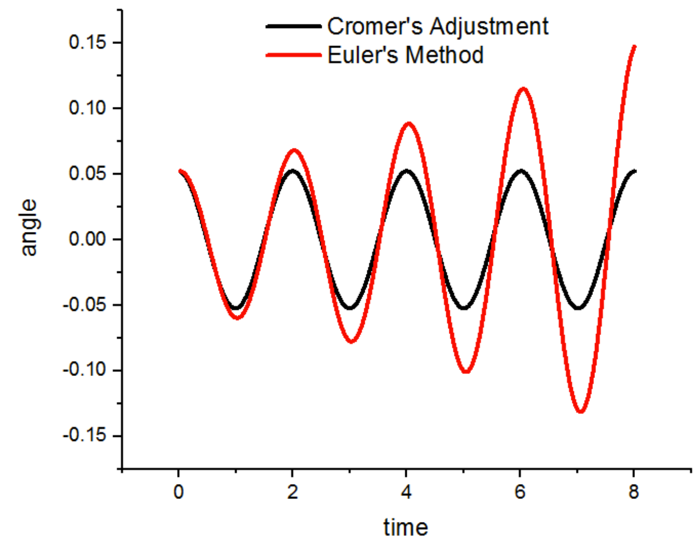
$$\theta_i = \theta_{i-1} + \omega_i * \Delta t$$

Another way to understand this situation is to try and combine Euler's Method's equations into one → you cannot do so without tangling up your order of operations, i.e., without having a current index rely on another current index.

$$Unsolvable\ in\ one\ step: \theta_i = \theta_{i-1} + \left(\omega_i + \frac{g}{\ell}\theta_{i-1} * \Delta t\right) * \Delta t$$

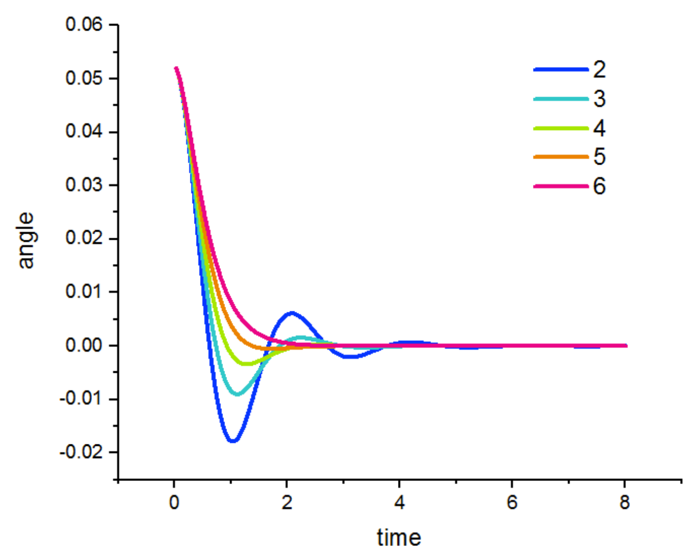
$$Solvable\ in\ one\ step: \theta_i = \theta_{i-1} + \left(\omega_{i-1} - \frac{g}{\ell}\theta_{i-1} * \Delta t\right) * \Delta t$$

Below we see Euler's pendulum accelerating over time, violating conservation of energy, whereas Cromer-Euler's pendulum acts as one would expect.



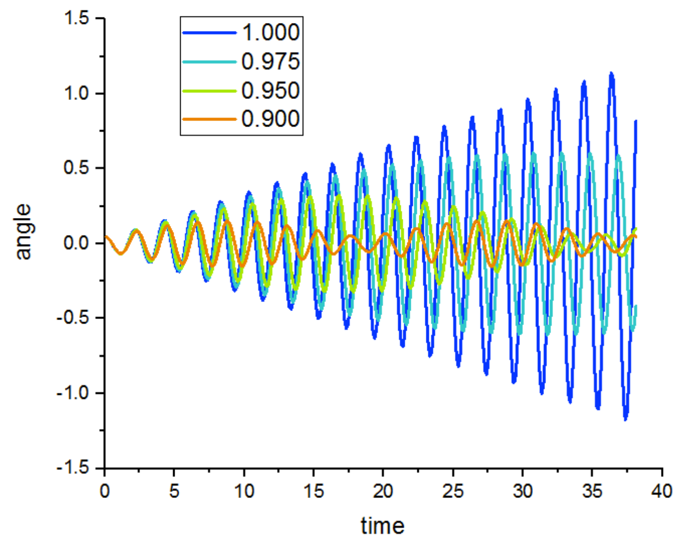
Angles are always in radians (initial angle θ_0 is henceforth always 3°), and time is seconds - however the scale is dependent on settings of dt

Now we add some spice by introducing a damping coefficient r - for resistance. This provides a more realistic model of the pendulum which won't oscillate forever.



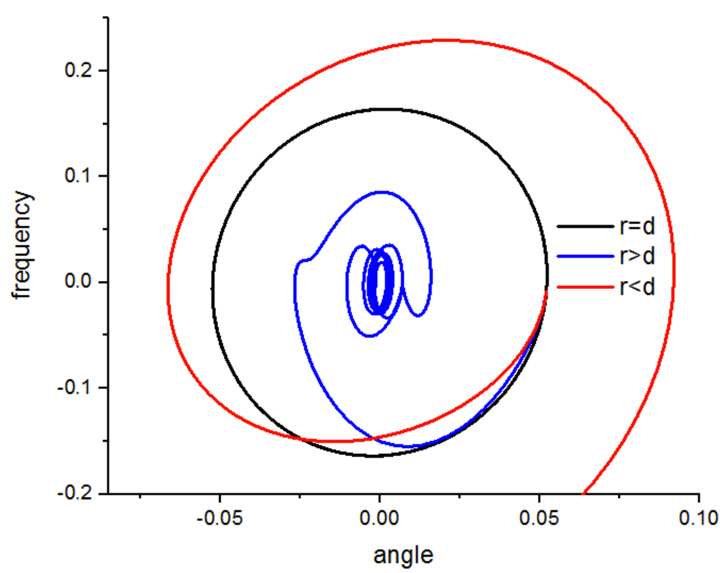
Various resistances imposed (r = 2, 3, 4, 5, 6), demonstrating under-damping (2-5) and critical damping (~6)

Finally we add a driving force d to the pendulum, which when combined with resistance can produce some surprising results. First however, we show what the push of a driving force can do – without resistance.



Driving force *d*, slowly approaching 1, demonstrating *resonance* with the natural force of the pendulum

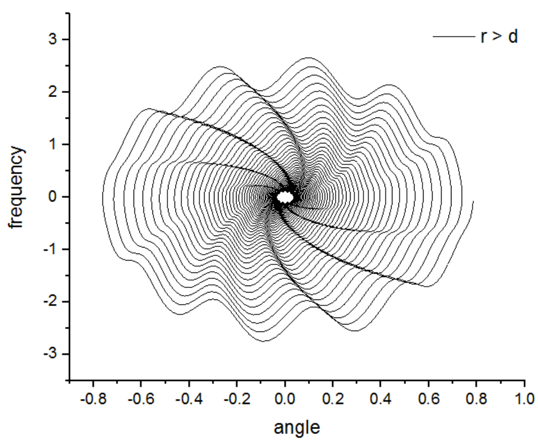
Above, we can see how slight adjustments can make a big difference. The canonical metaphor here is that of pushing someone on a swing, and timing your pushes to match up with the natural force we calculated earlier, imposed by gravity. What you find is that *at resonance* the driving force will magnify the motion linearly. This effect can be considered the opposite to *critical* damping – when the motion is halted most efficiently with equal and opposite force to F_θ (slowing someone down to a stop on a swing within half a period). Now for the fun part!

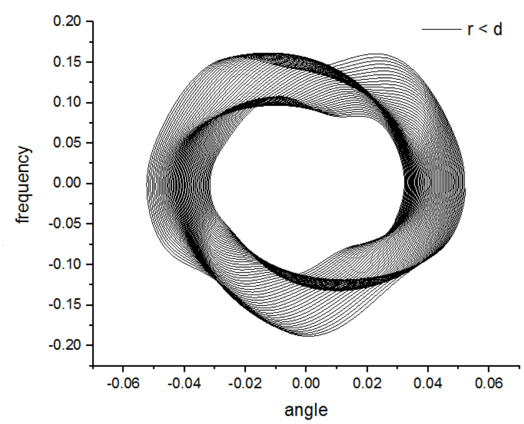


"Phase diagram" showing frequency vs angle. Here we have three situations (r=d) - forming a perpetual oscillation; (r>d) - the motion is slowed eventually to a halt; and (r<d) where the motion is unbounded

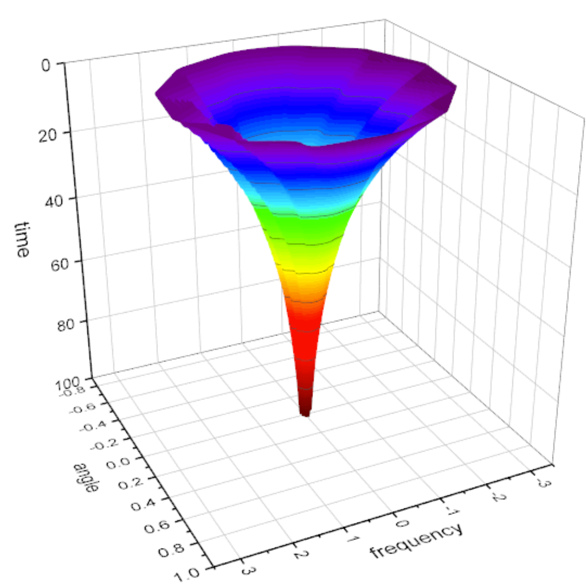
The dramatic wiggle in the above diagram is what we call *chaotic motion*, because it's notoriously difficult to predict or model. This chaos is a consequence of the way the driving force and resistance both affect the oscillation simultaneously. If r and d are not neat ratios of one another, e.g., $r = 0.67, d = 1.92$, then you will always see some chaos in the motion. One can think of this as something akin to musical dissonance, which we don't like to hear because it's difficult for our brains to process the irrational ratios.

Below are two plots of somewhat harmonious motion, where the values of r and d divide each other well, thus creating a superposition of motion with new symmetries. Dare I say, these plots are quite pretty.





And finally, because these plots seem to beg for a 3rd dimension, I've plotted the phase diagram over time. Here we see the motion slow to a halt at the bottom.



Below is a C code implementation, which produced the plots.

```
//Johnathan von der Heyde - 2019
//Modeling Simple Pendulum with Cromer-Euler Method

#include <stdio.h>
#include <math.h>
#define PI 3.1415927
#define g 9.8                  //gravitation (m/s^2)
#define l 1.0                  //pendulum length (m)

int main() {

        double omega, theta, r, d; //frequency, angle, resistance, and driving force
        double f = pow(g/l, 0.5);  //natural frequency
        double T = 2.0*PI/f;       //natural period
        double A = 1;              //visual amplitude
        double t = 0;              //time
        double dt = 0.01;          //time step – can be configured to period

        printf("Enter theta (degrees)\n");
        scanf("%lf", &theta); theta *= (PI/180);
        printf("Enter r (resistance)\n");
        scanf("%lf", &r);
        printf("Enter d (driving force)\n");
        scanf("%lf", &d); d *= f;

        FILE * file = fopen("OUT", "w");    //main loop can be configured for finite
                while (t < 500) {           //time extension, or when pendulum stops

                        t+=dt;
                        omega -= ((g*theta/l)+(r*omega*(-A*cos(d*t))))*dt;
                        theta += omega*dt;
                        fprintf(file, "%f\t%f\t%f\n", theta, omega, t);
                }
        fclose(file);
        return 0;
}
//END
```