Coding Physics

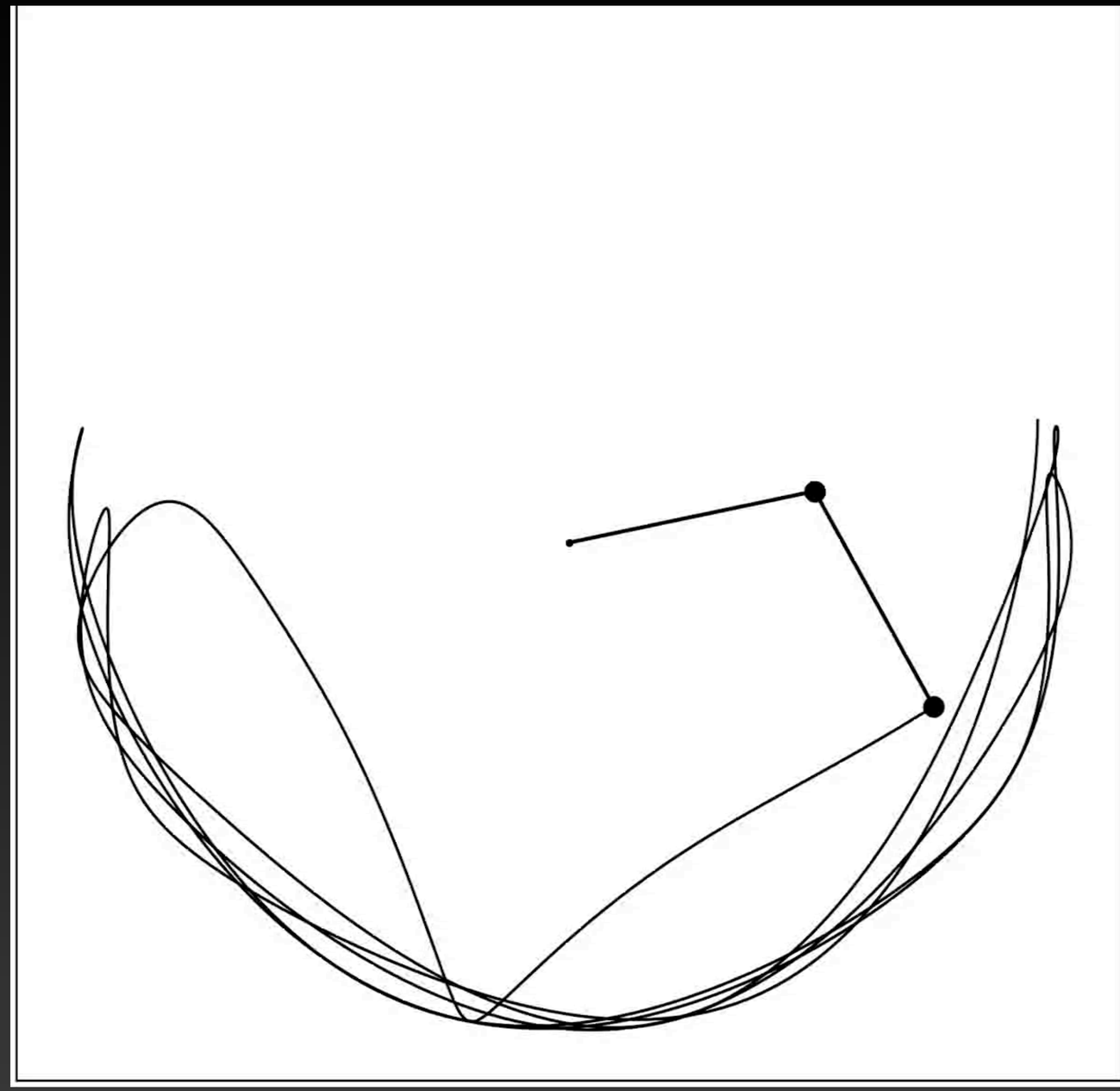
The Motion of Pendulums using Computers

Garett Brown 2021

UNIVERSITY OF TORONTO

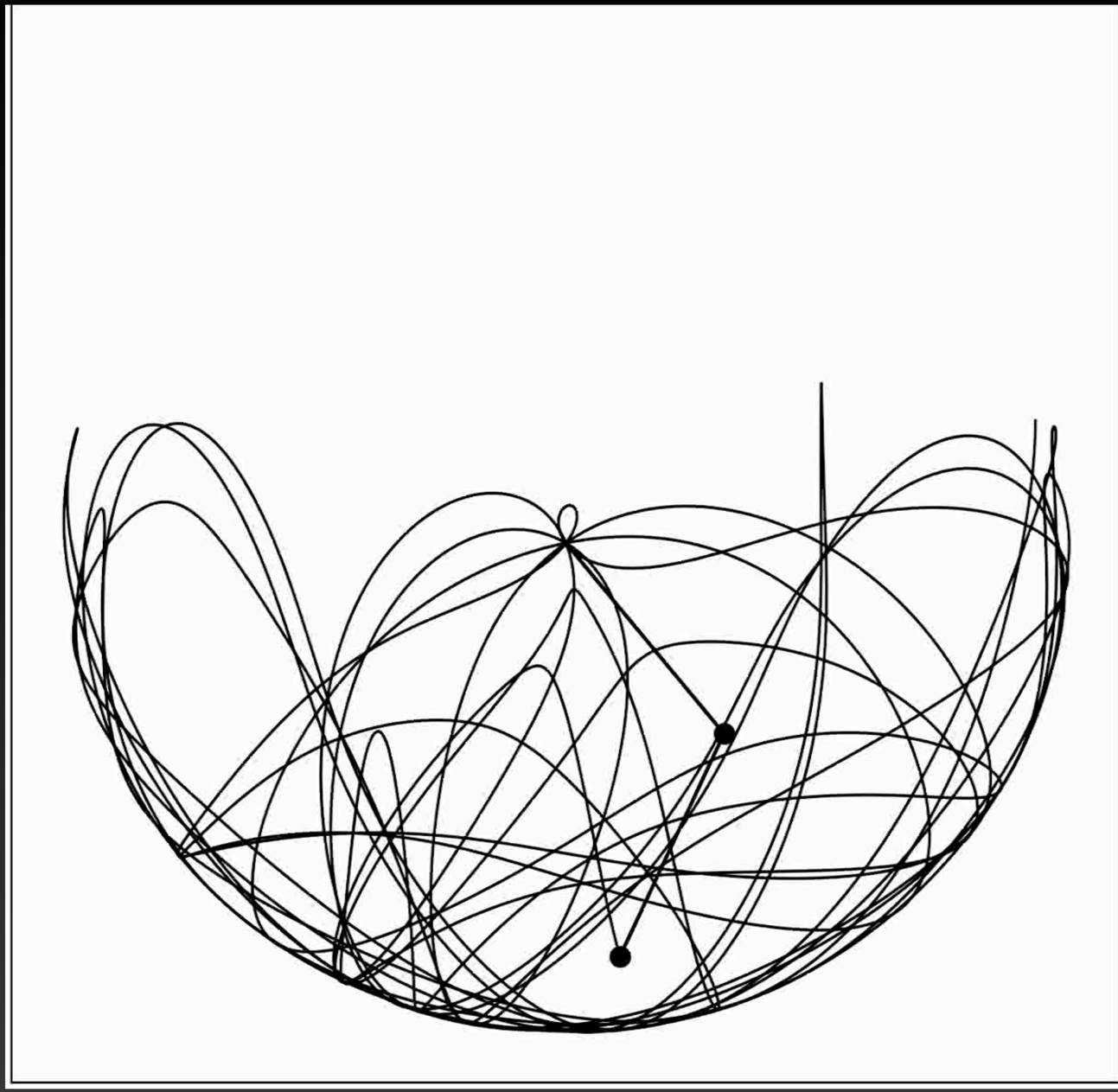
Why should we solve Physics Problems with Computer Code?

- Some experiments cannot be done in a physical lab.
  - Simulating the universe or the planets.
- Some experiments are too costly to practically run thousands of times.
- Machine Learning methods and AI can help us see new patterns



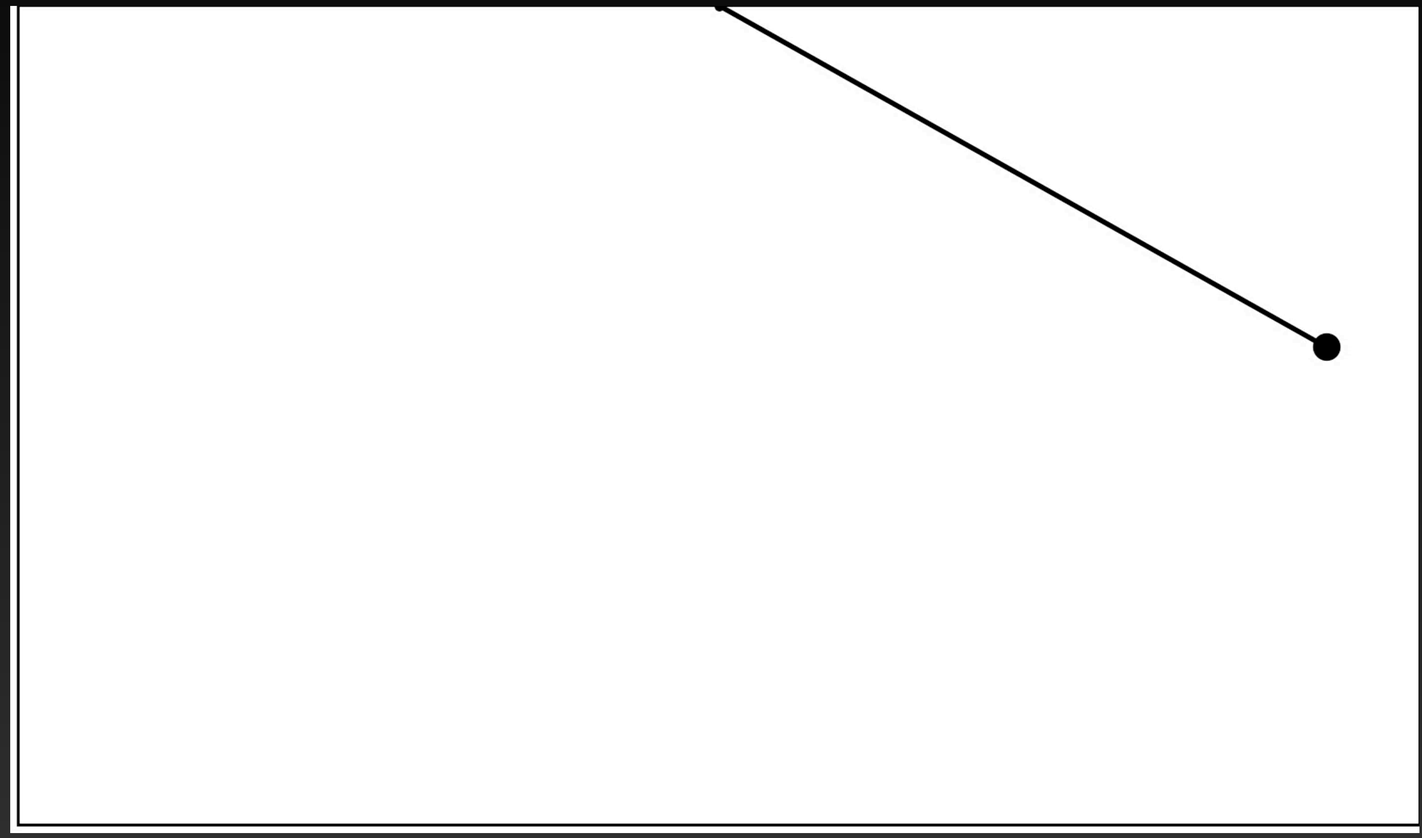
How do we solve Physics Problems with Computer Code?

- First, we need to be able to describe the motion.
- For many problems this can be done with Newton's 2nd law.
  - $F = ma$ and $\tau = I\alpha$
- We will go through how to do this for the simple pendulum



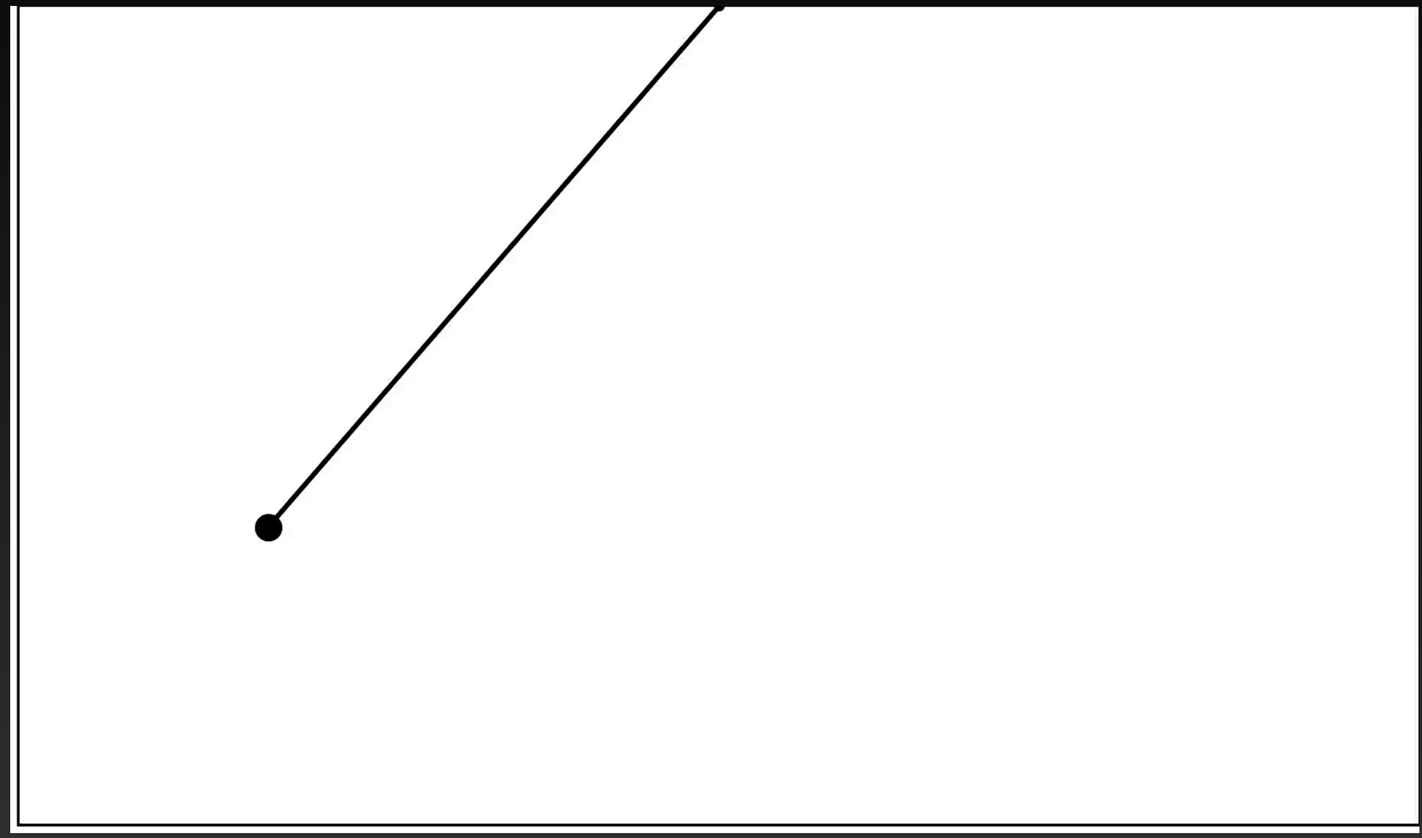
The Pendulum

- The simple pendulum has many interesting properties that make it useful to study.
- As such, it is often used to model many other physical phenomena, and is used by analogy to describe many more.



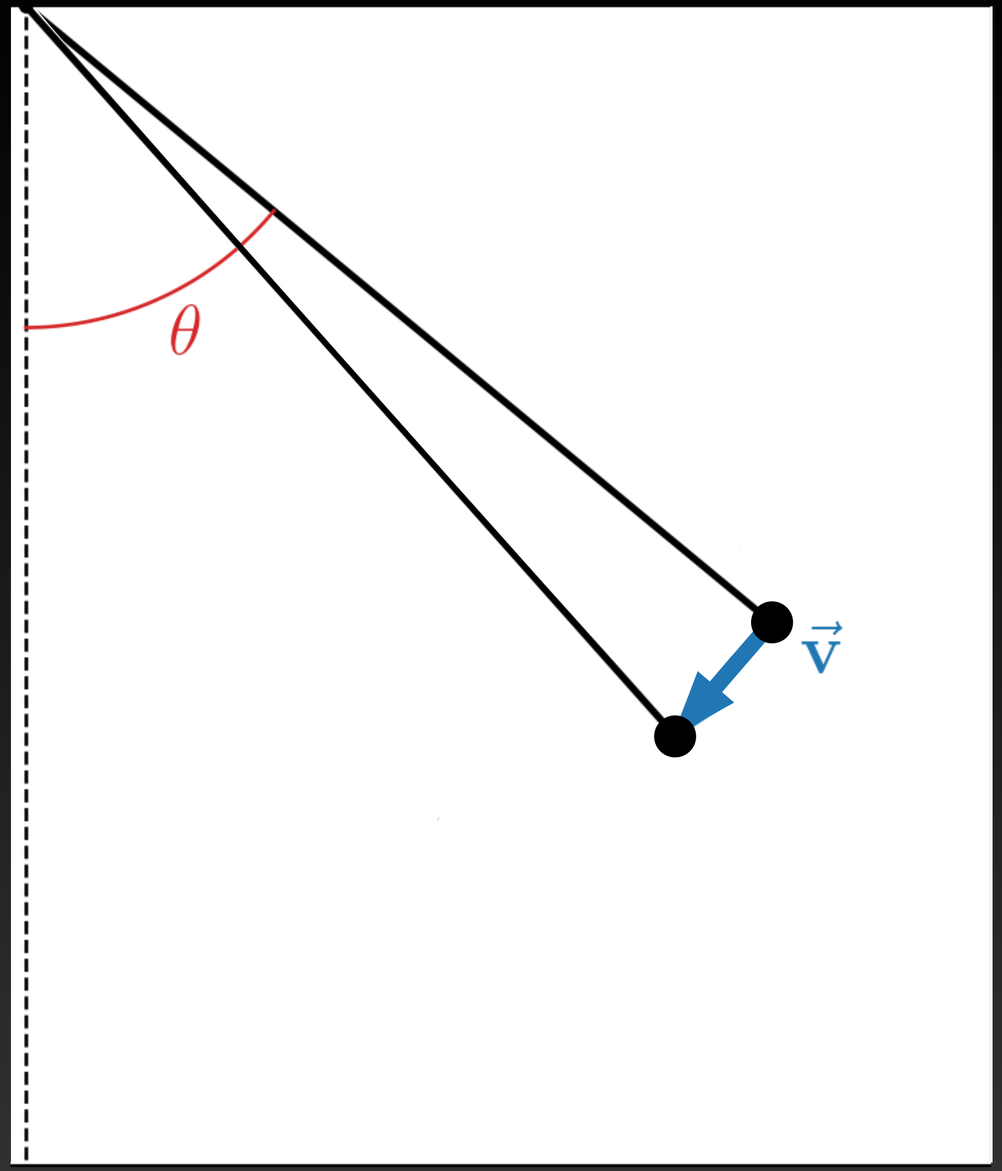
How do we solve Physics Problems with Computer Code?

- If we know the location (position) of an object, how can we know where it will go next?
- We can determine where it will be next if we know the velocity of the object.



How do we solve Physics Problems with Computer Code?

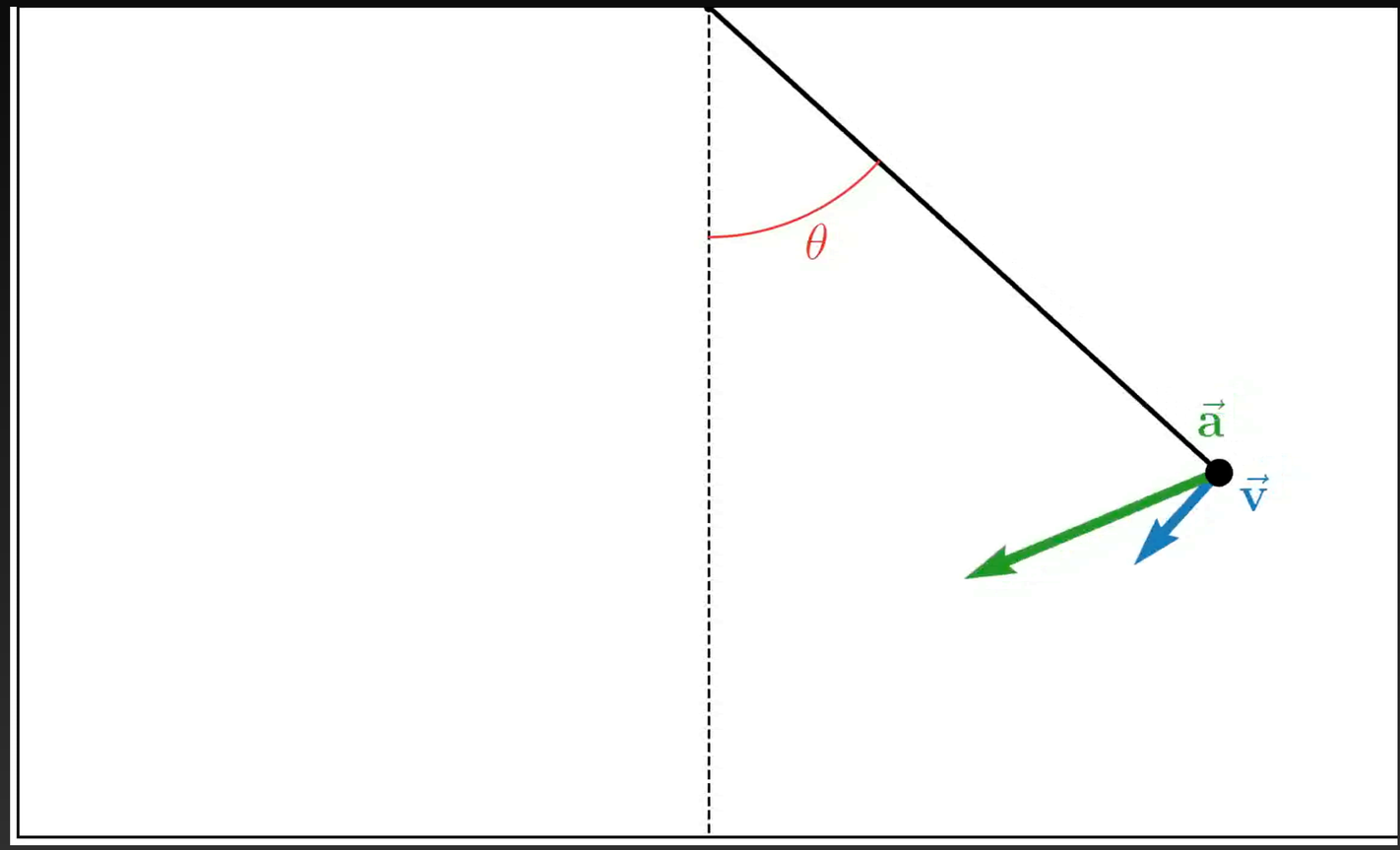
- The position changes as the object moves.
- Velocity is the change in position over the change in time.
  - $$v = \frac{x_f - x_i}{t_f - t_i} = \frac{\Delta x}{\Delta t}$$
- We can use this information to calculate where it will be next.
  - $\Delta x = v\Delta t$
- But what if the velocity is changing too?



Δ is the Greek letter “Delta”

How do we solve Physics Problems with Computer Code?

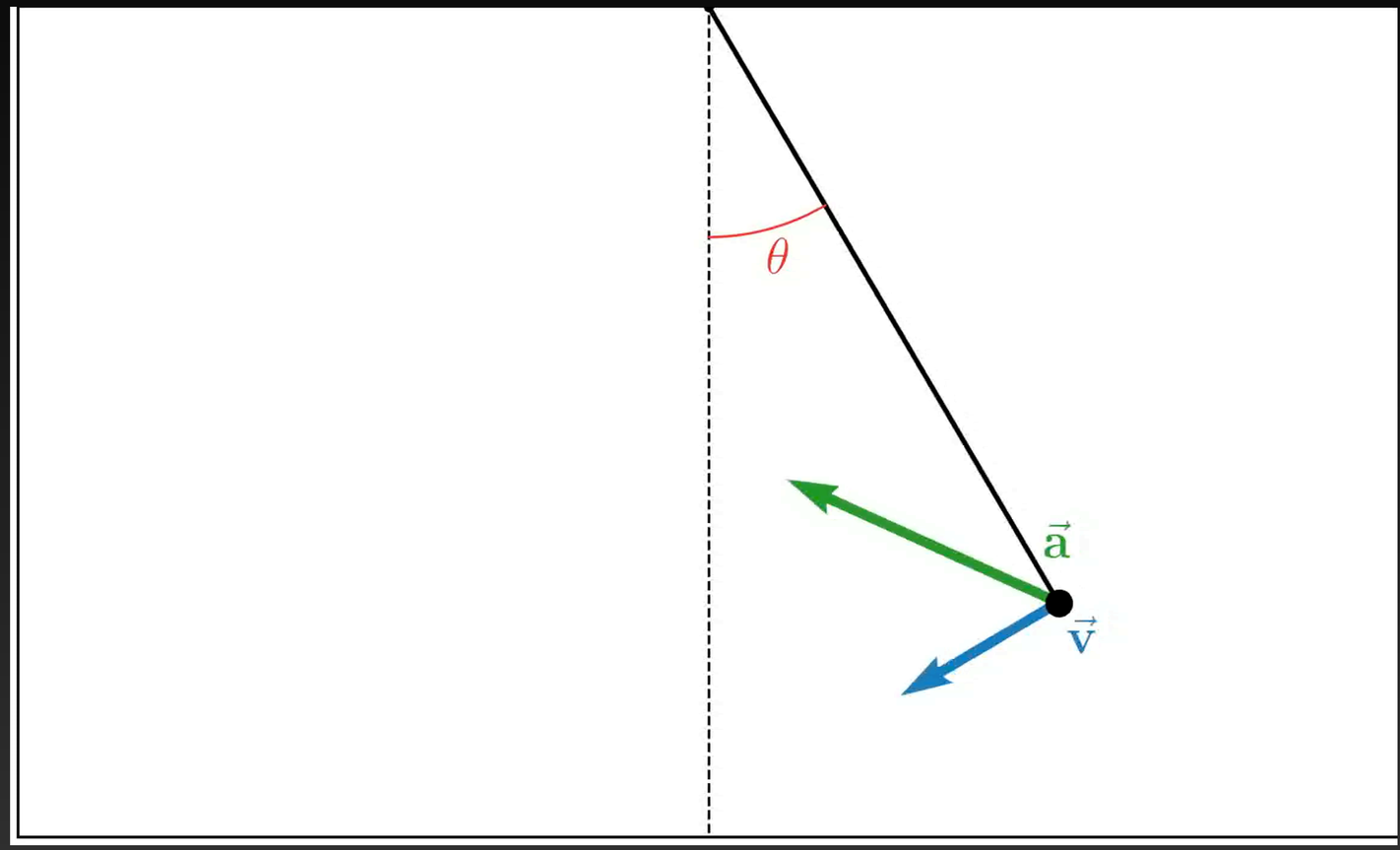
- If the velocity is also changing, then we can determine how the velocity changes based on the acceleration.
- This is because acceleration is the change in velocity over the change in time.
- $$a = \frac{v_f - v_i}{t_f - t_i} = \frac{\Delta v}{\Delta t}$$



Δ is the Greek letter “Delta”
Often used to mean “change in” or “difference”

How do we solve Physics Problems with Computer Code?

- But we come full circle.
- The change in position tells us the velocity and the change in velocity tells us the acceleration, but we don’t know where the object will be unless we first know the velocity and the acceleration.
- Thus, we use Newton’s 2nd Law of motion to first determine the acceleration of the object.

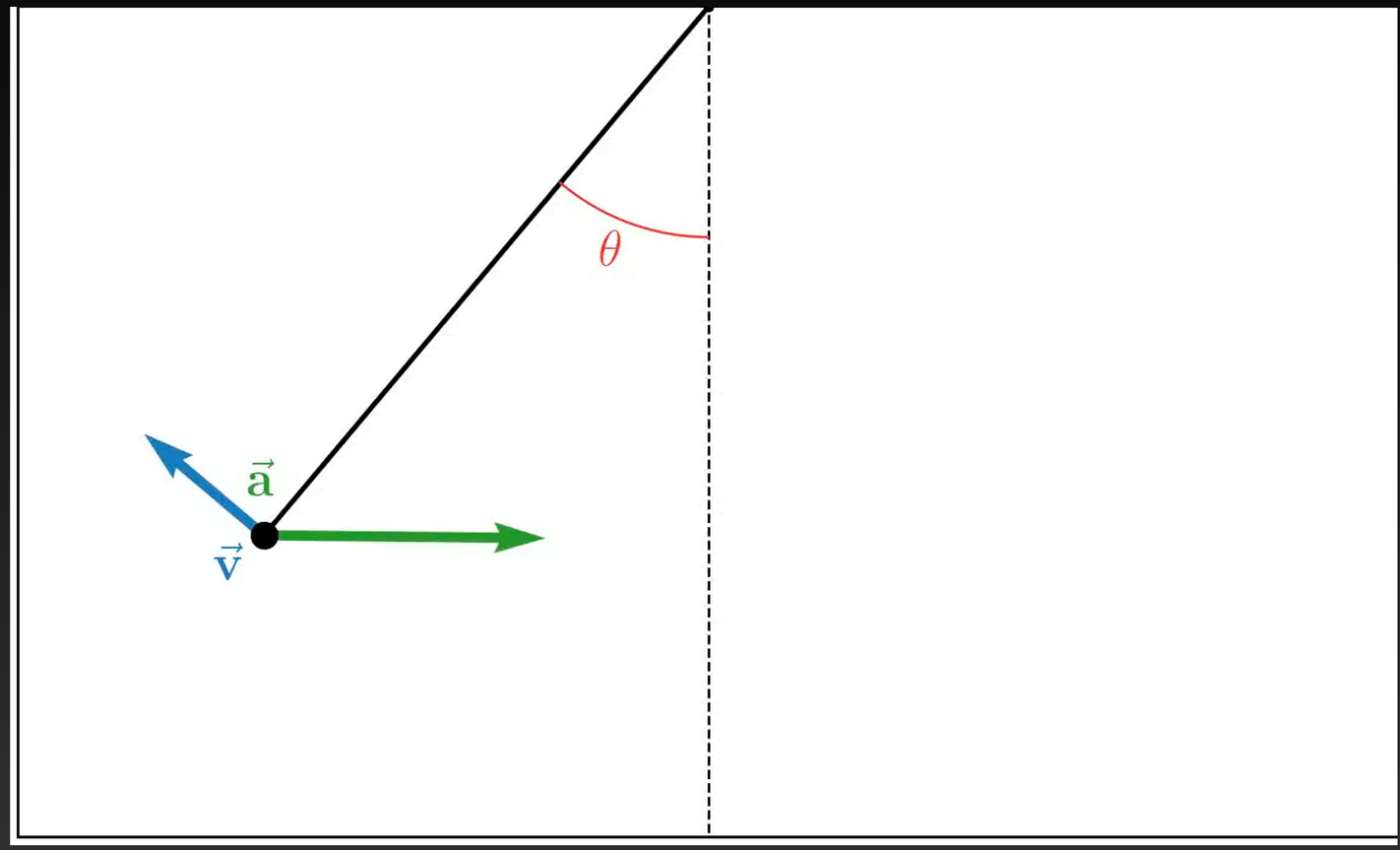


The Pendulum

Follow Along:

- Python: https://trinket.io/library/trinkets/e455871d49
- JavaScript: https://editor.p5js.org/zyrxvo/sketches/VNttrXjWP

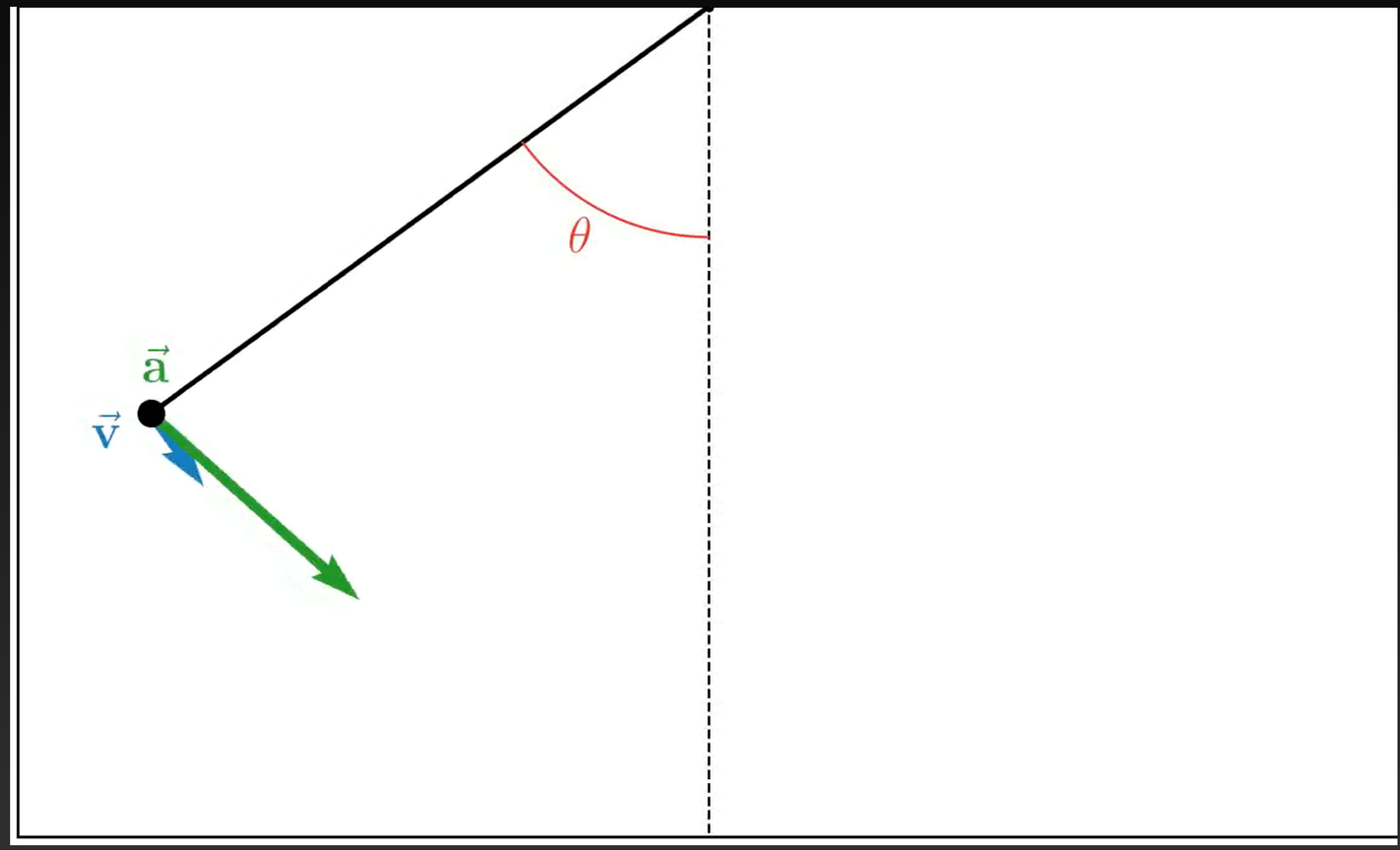
As we go through the physics, fill in what α, ω, and θ are equal to in the ** Code Solution Here: ** section.



α is the Greek letter “alpha”, ω is the Greek letter “omega”, and θ is the Greek letter “theta”

The Pendulum

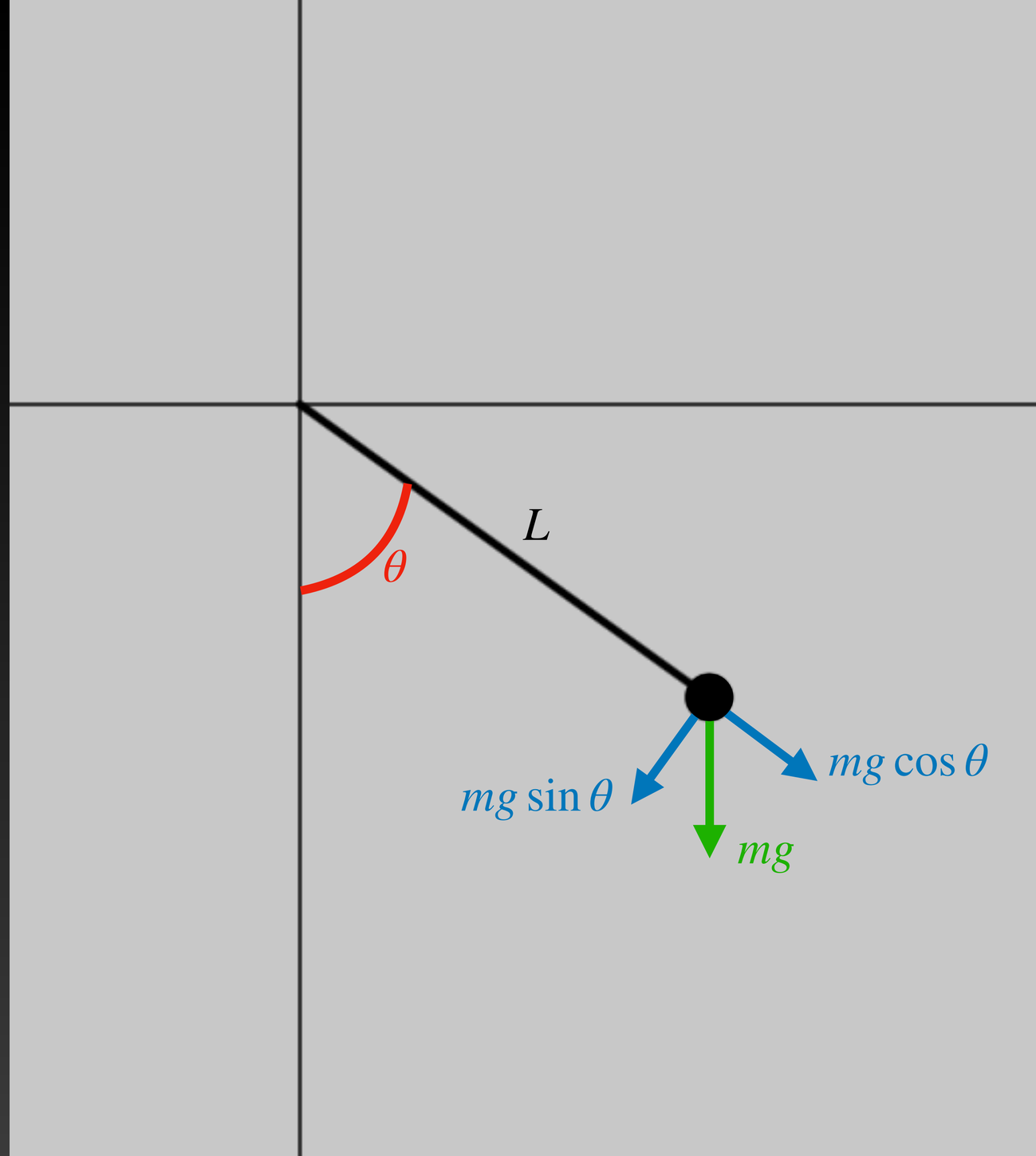
- We begin with Newton's 2nd law for rotational systems.
- Often this is written as:
  - $\tau = I\alpha$
- where τ is the torque, I is the moment of inertia, and α is the angular acceleration.



τ is the Greek letter "tau"

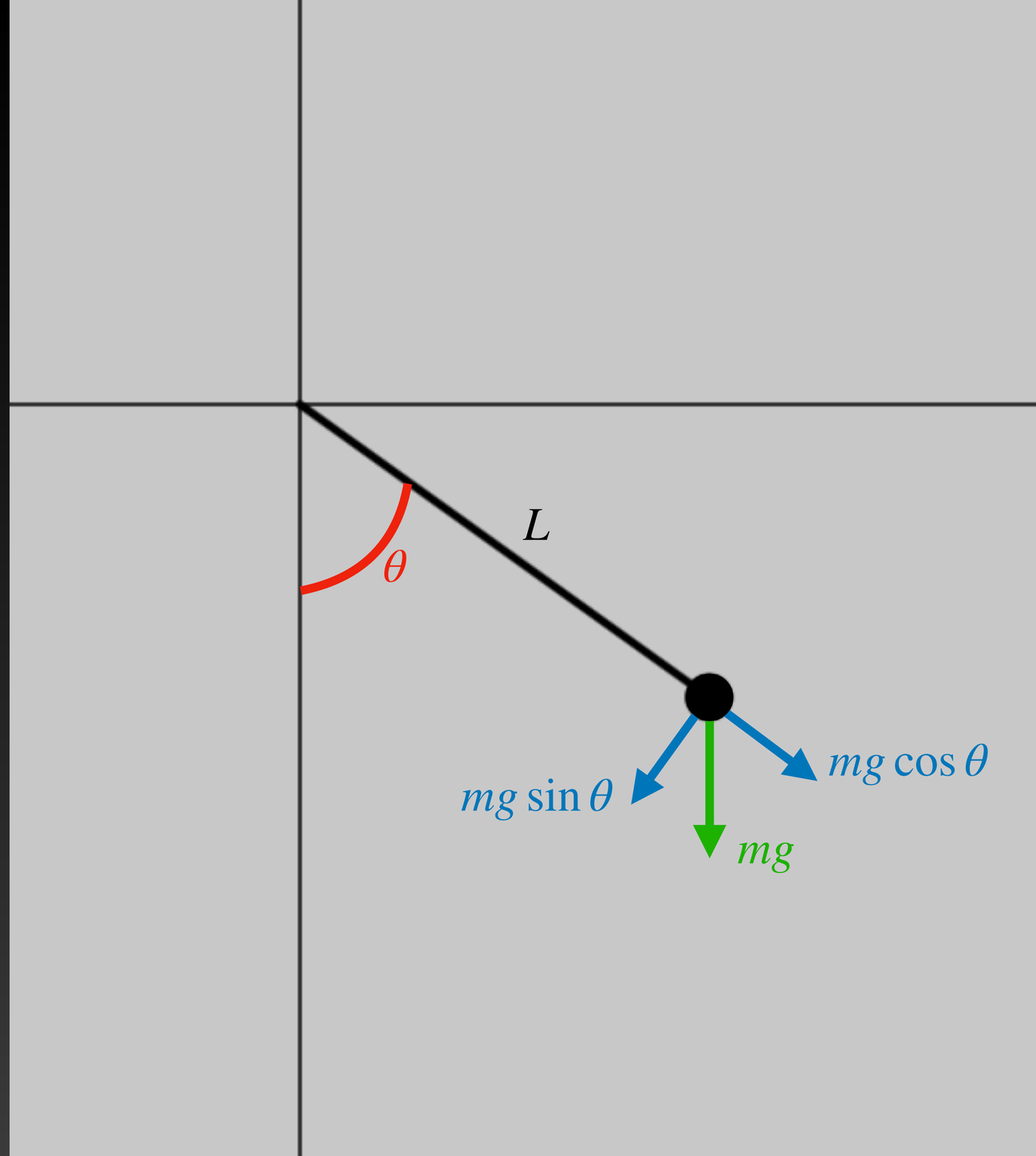
The Pendulum

- The position of the mass is determined only by θ, the angle measured from the vertical axis.
- Gravity is the only force causing a torque on the mass, which is given by $\tau = -mgL\sin\theta$.
- The moment of inertia for the mass is $I = mL^2$
- The angular acceleration of the mass is simply α.



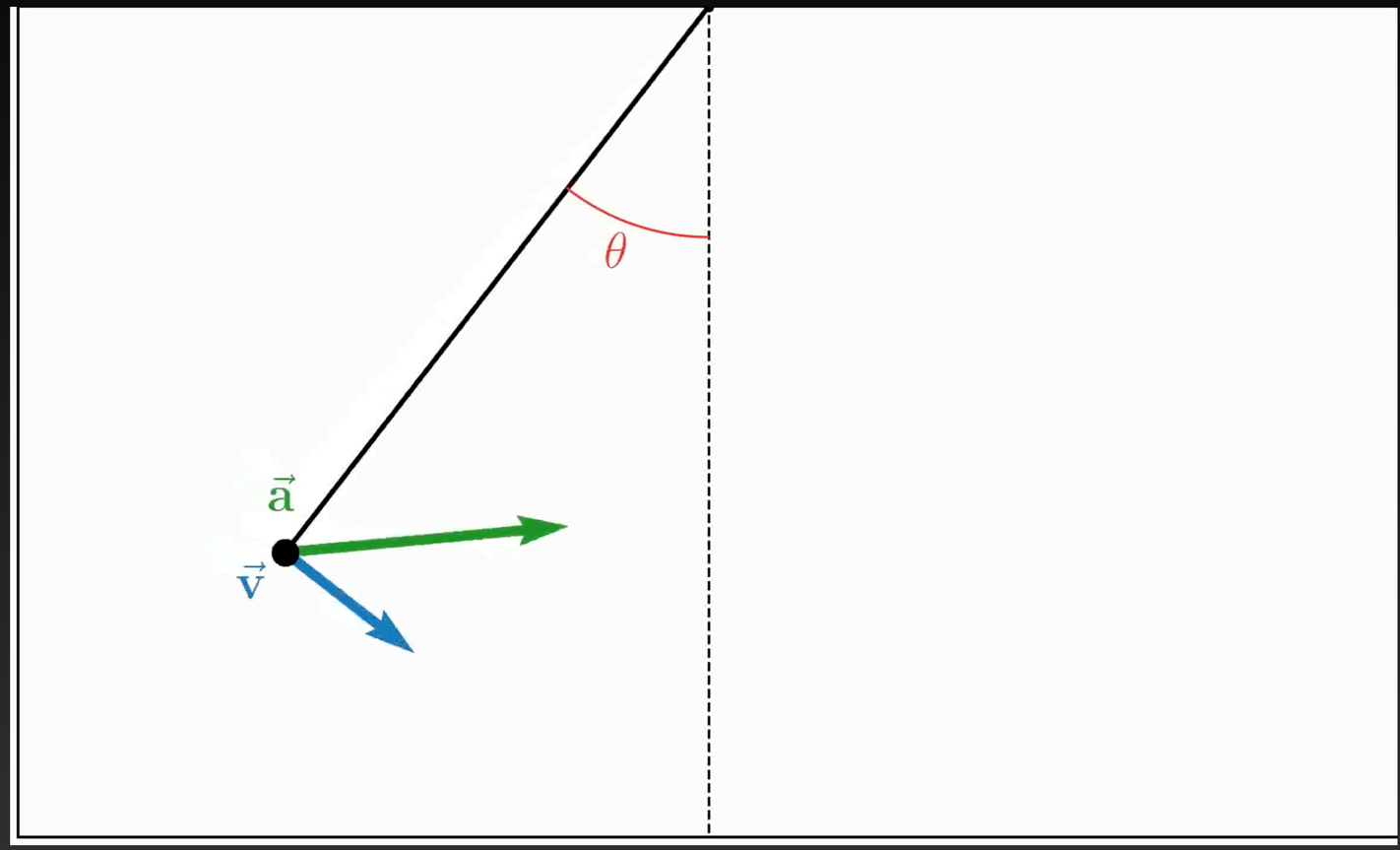
The Pendulum

- Bringing this all together gives,
  - $\tau = I\alpha$
  - $-mgL\sin\theta = mL^2\alpha$
- Now rearrange the equation to better see how the angular acceleration (α) is related to the angular position (θ).
  - $\alpha = -\frac{g}{L}\sin\theta$



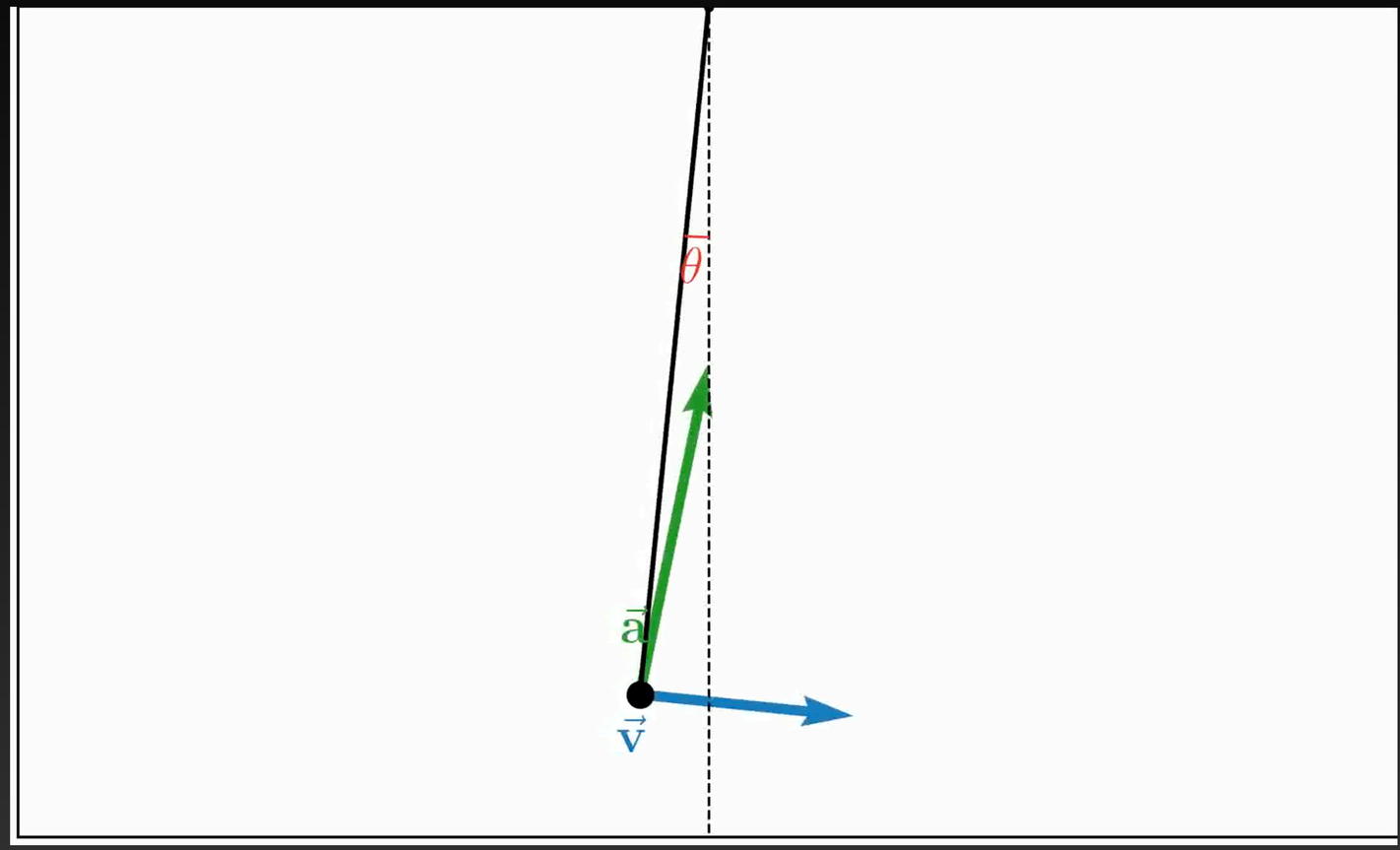
Simple Harmonic Oscillator

- Now we can update the angular velocity (ω) based on the calculated angular acceleration.
  - Using $\alpha = \Delta\omega/\Delta t$, we can rearrange the equation for ω_f
    - $\Delta\omega = \alpha\Delta t$
    - $\omega_f - \omega_i = \alpha\Delta t$
    - $\omega_f = \omega_i + \alpha\Delta t$



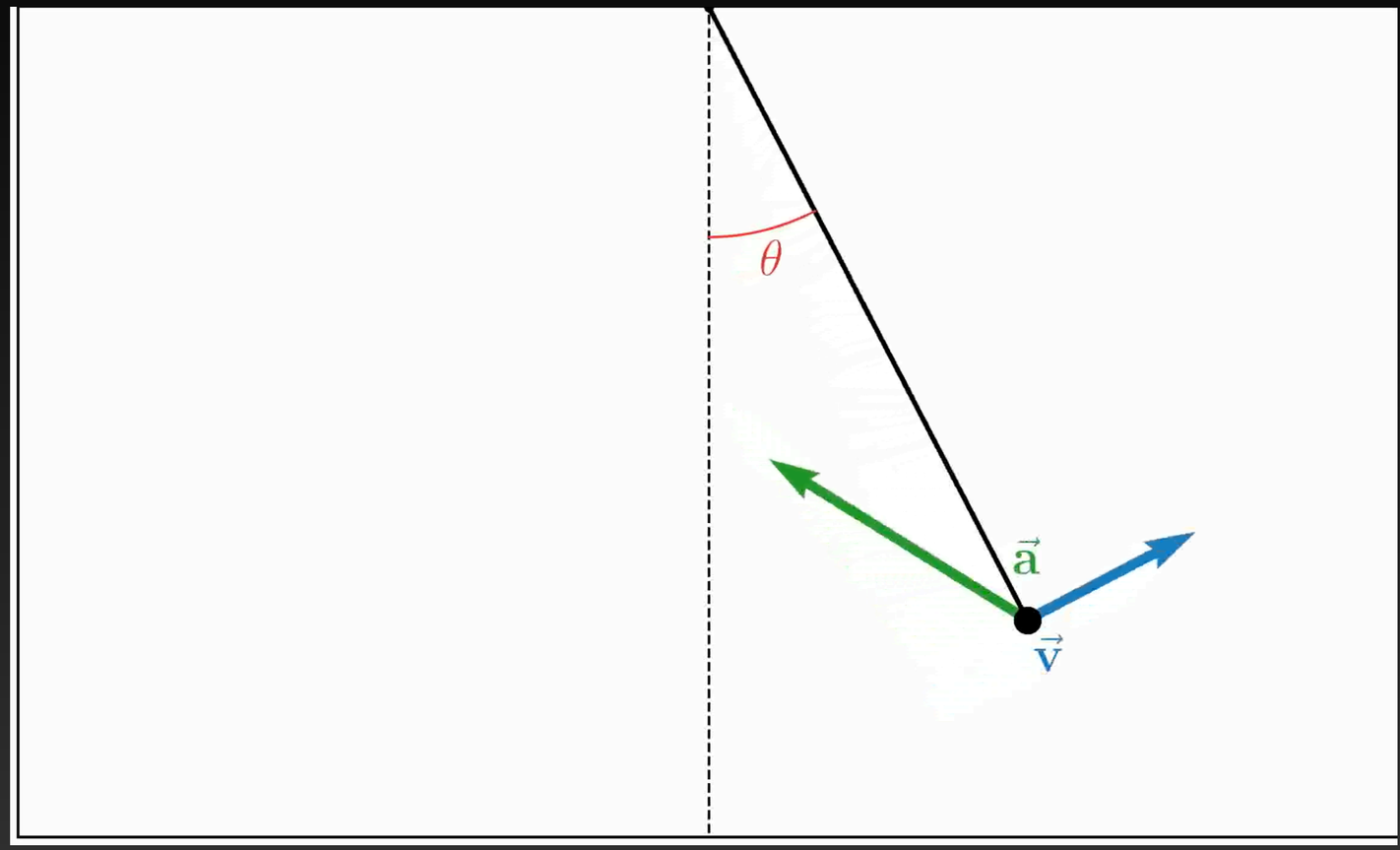
Simple Harmonic Oscillator

- Now we can update the position based on the updated velocity.
  - Using $\omega = \Delta\theta/\Delta t$ we can rearrange the equation for x_f
    - $\Delta\theta = \omega\Delta t$
    - $\theta_f - \theta_i = \omega\Delta t$
    - $\theta_f = \theta_i + \omega\Delta t$



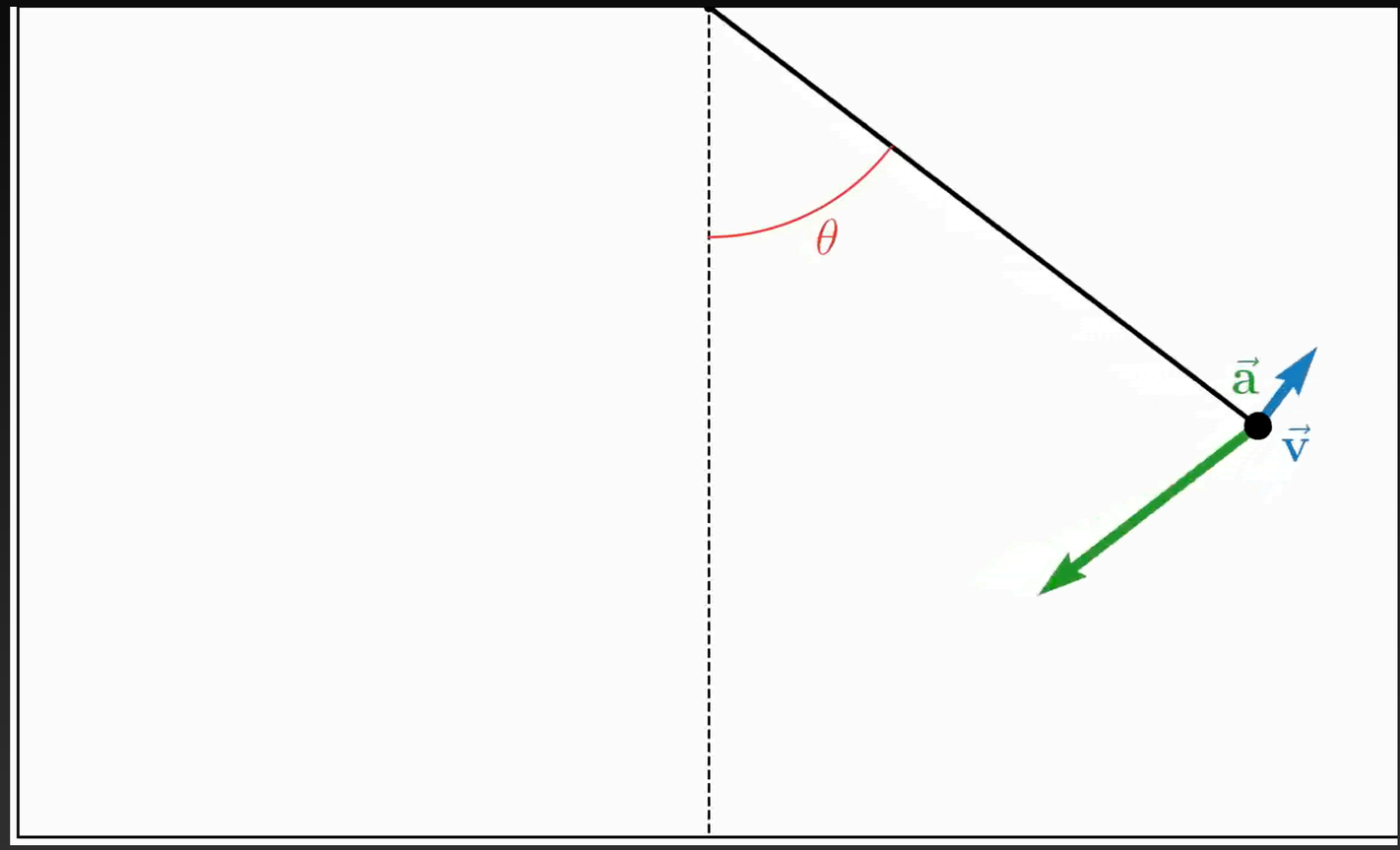
The Pendulum

- Putting this all together we have,
  - $\alpha = -\frac{g}{L}\sin\theta$
  - $\omega_f = \omega_i + \alpha\Delta t$
  - $\theta_f = \theta_i + \omega\Delta t$
- If we calculate these 3 equations over and over again we can animate the position of the mass over time.



The Pendulum

- In programming, the expression on the right-hand side of the equation is assigned to the left hand side.
- Thus, the initial angular position and velocity can be used to calculate the final values, so the code looks more like this:
  - $\alpha = -(g/L)\ \sin\theta$
  - $\omega = \omega + \alpha\Delta t$
  - $\theta = \theta + \omega\Delta t$



The Pendulum
Coding Time!

- For JavaScript go to https://editor.p5js.org/zyrxvo/sketches/VNttrXjWP
- For Python go to https://trinket.io/library/trinkets/e455871d49
- All of the code used to animate the pendulum is already there. Just find the part that says,

```
**********************************
      Code Solution Here:
**********************************



**********************************
```

$$\alpha = -(g/L)\sin\theta$$

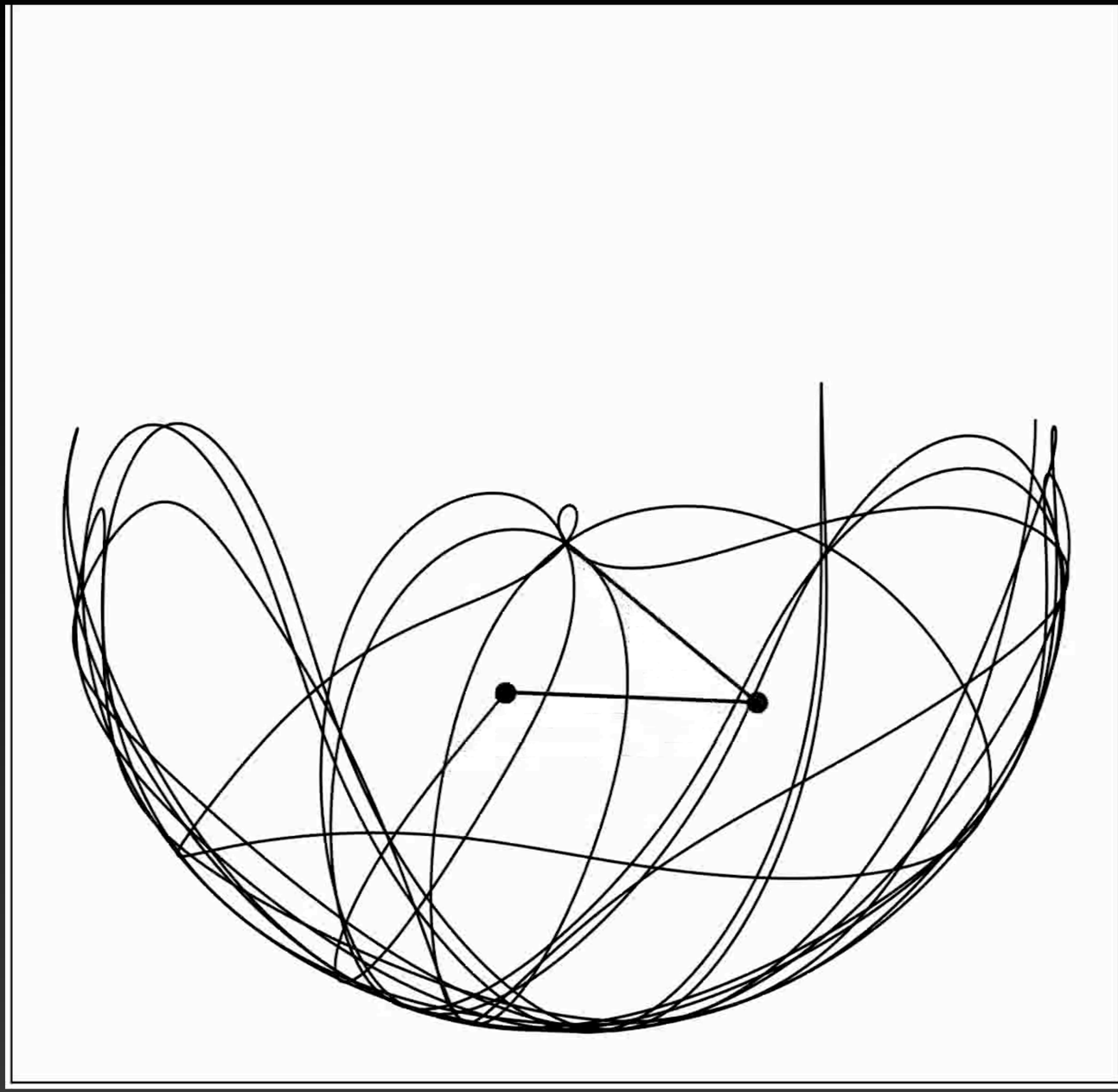
$$\omega = \omega + \alpha\Delta t$$

$$\theta = \theta + \omega\Delta t$$

- and code up your solution between the lines. When you think you've got it, hit the play button at the top of the page and see if it worked! You can also play around with the initial conditions and physical constants.
- If you get stuck, the solutions can be found at
  - JavaScript: https://editor.p5js.org/zyrxvo/sketches/mvERTFb5R
  - Python: https://trinket.io/library/trinkets/77ae8a6382
- As a bonus, see if you can add damping!

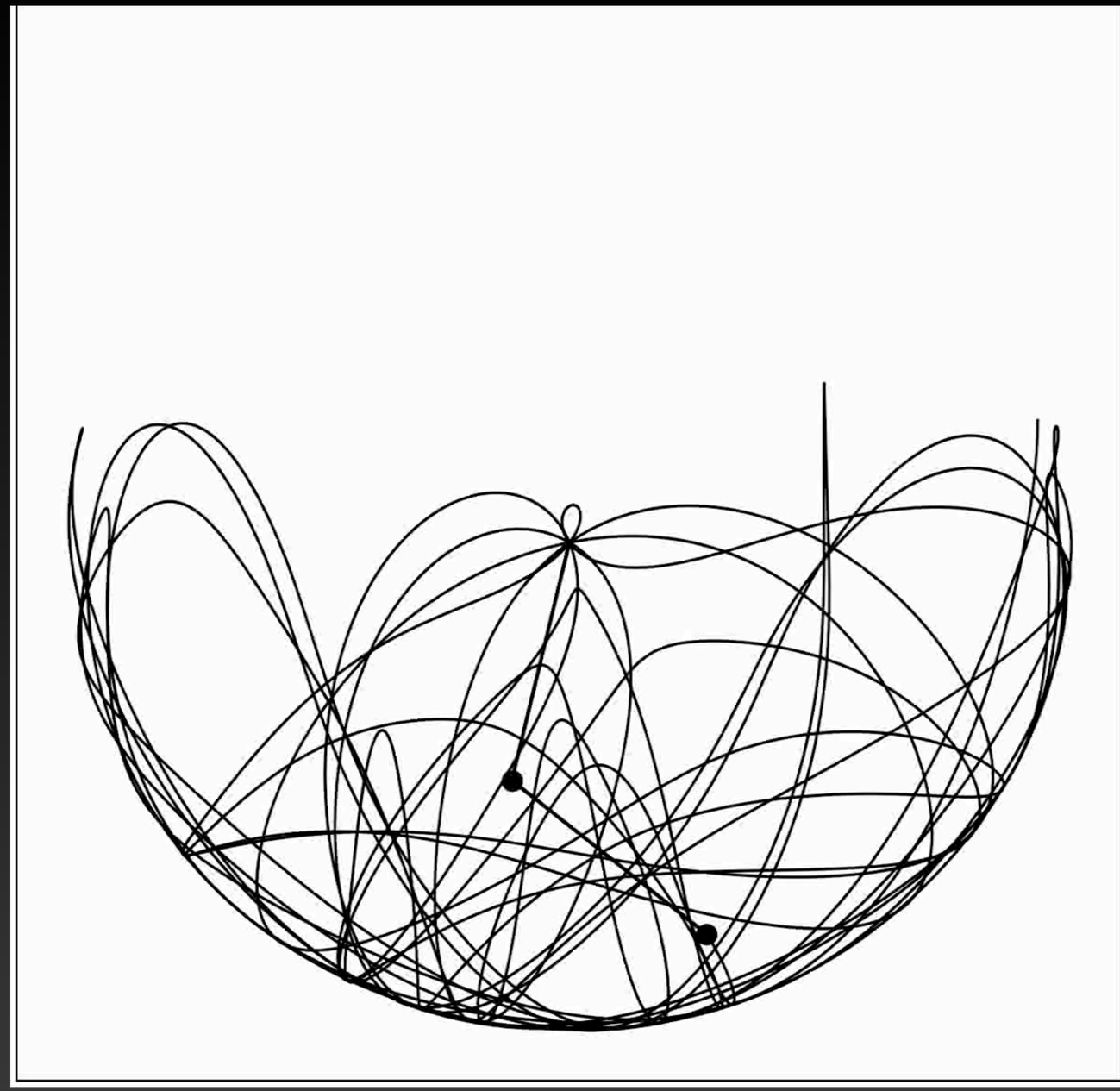
The Double Pendulum

- We are not going to derive the equations of motion for this one, instead we're going to spend some time exploring and tinkering.
- JavaScript: https://editor.p5js.org/zyrxvo/sketches/VpCiCDQ5s
- Python: https://trinket.io/library/trinkets/f0de22daa9



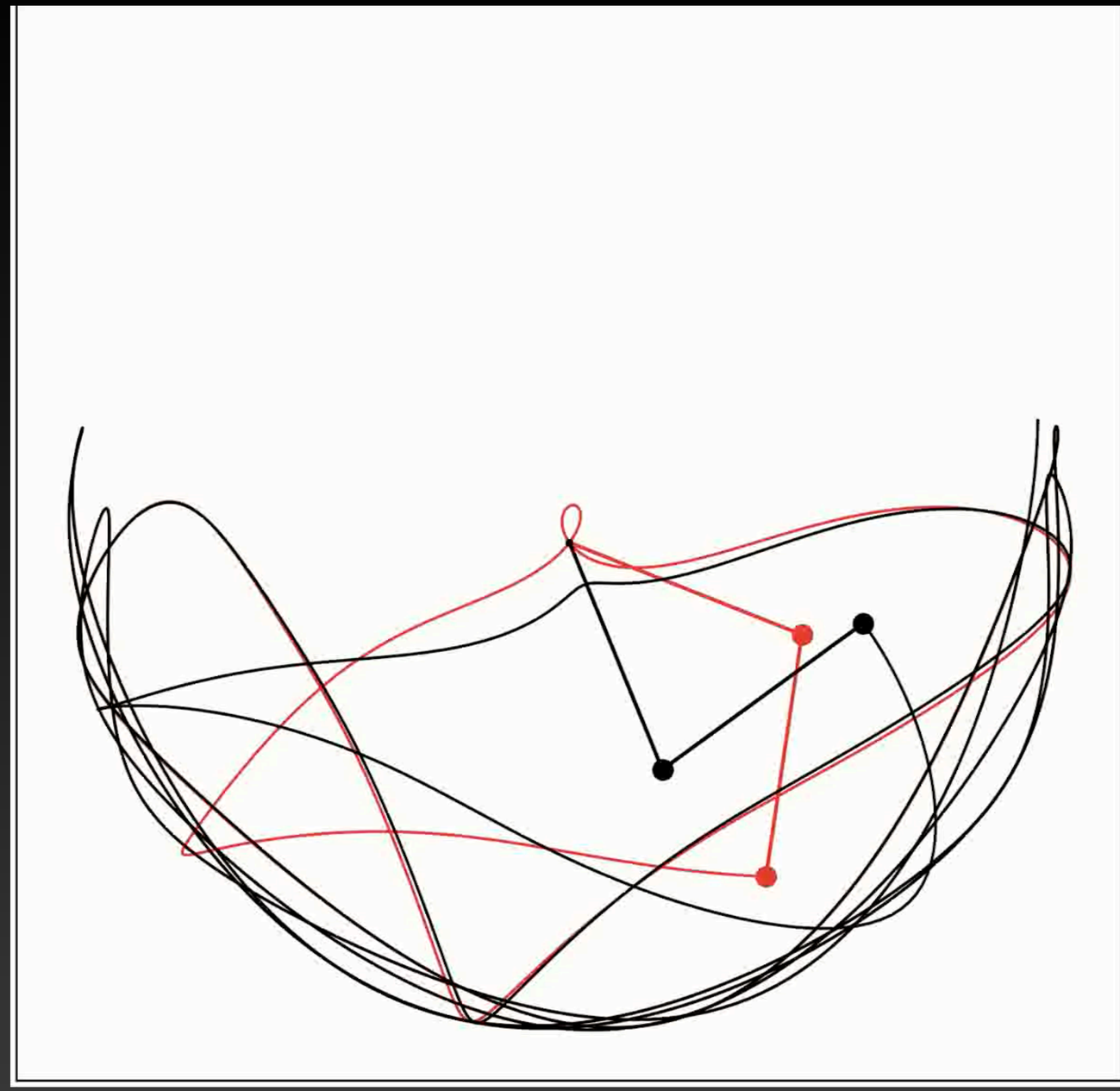
The Double Pendulum

- Play around with the initial conditions.
  - What happens if you start it so that both pendula are straight up?
  - Are you able to find any interesting behaviours?
  - Can you find a way to make the system numerically unstable? (Hint: Try increasing the velocity.)



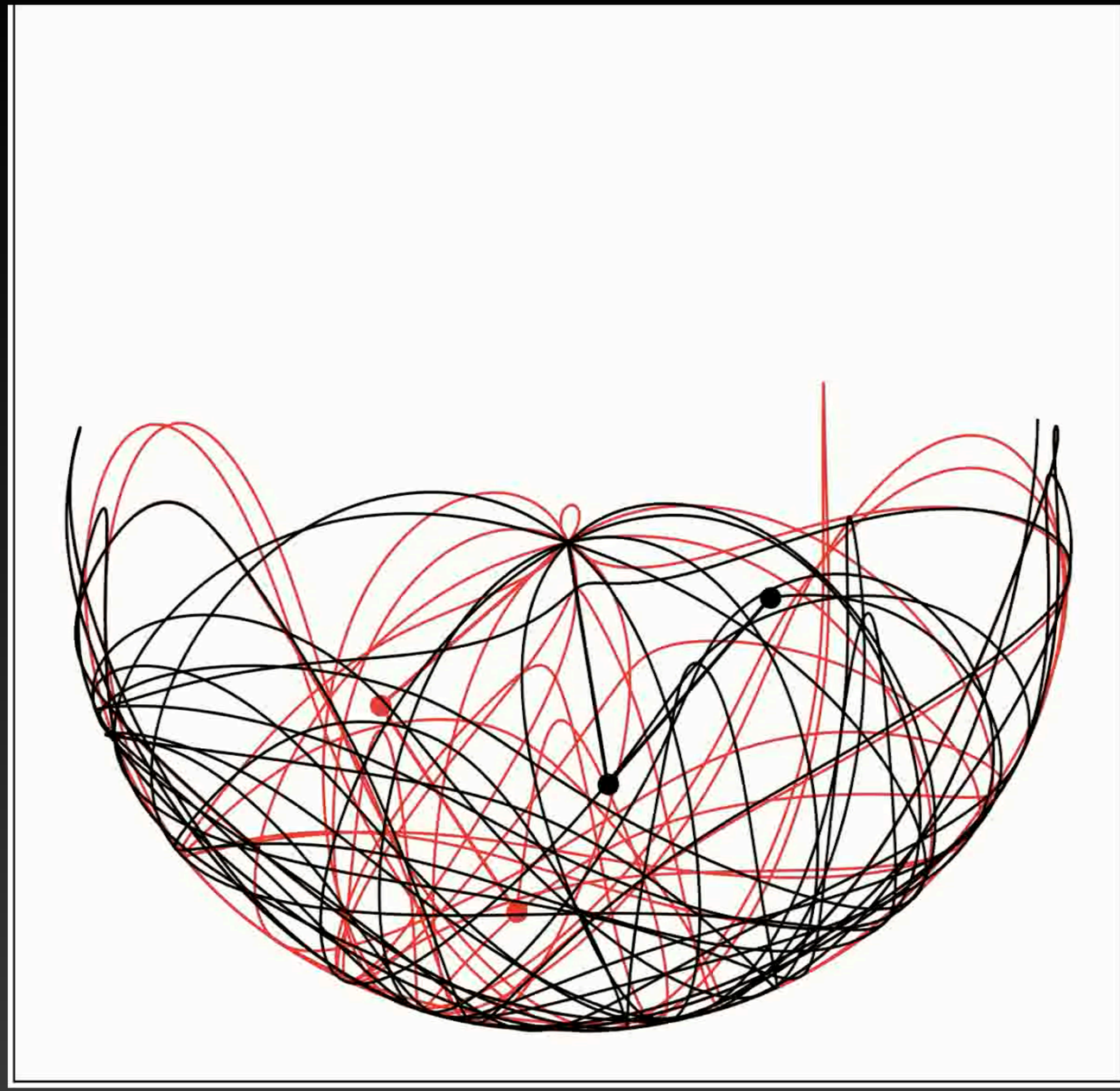
Chaos

- Small changes to the initial conditions can give very different results.
- The motion is determined, but not unpredictable.
- Small differences in initial conditions, such as errors in measurements or rounding errors in numerical computation, can yield widely diverging outcomes.



Chaos

- If we start the Double Pendulum with two almost identical initial conditions, after a short time their paths diverge. Code to play around with these can be found at
- JavaScript: https://editor.p5js.org/zyrxvo/sketches/cobz6b4Y7
- Python: https://trinket.io/library/trinkets/065cd7ab36
- MinuteLabs: http://labs.minutelabs.io/Chaotic-Pendulum/



Thank you
Happy Coding

UNIVERSITY OF
TORONTO