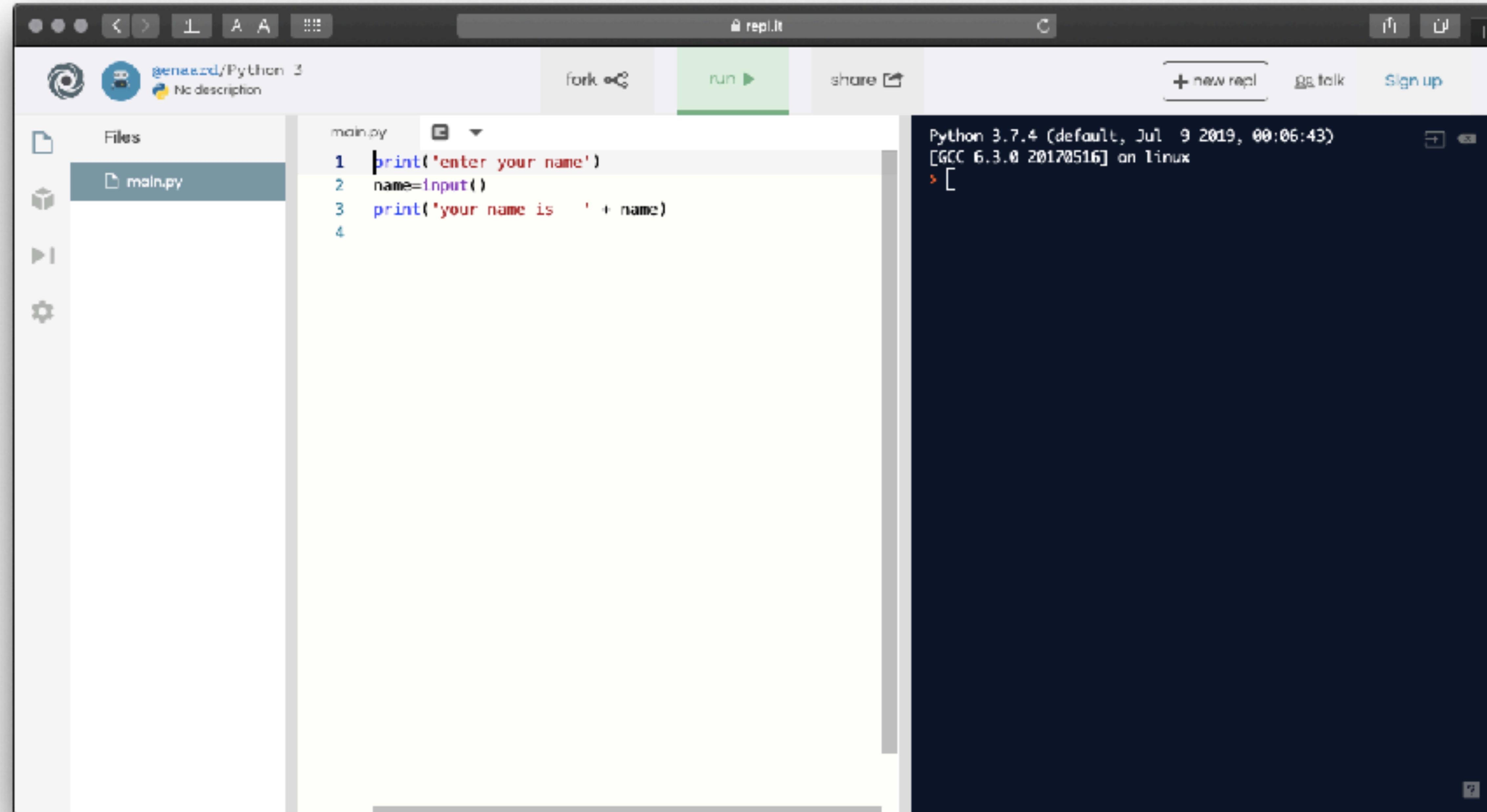
Numerical Modeling with Python

Harmonic Motion

Dr Matthew Edmonds

You can try the examples in today's lecture

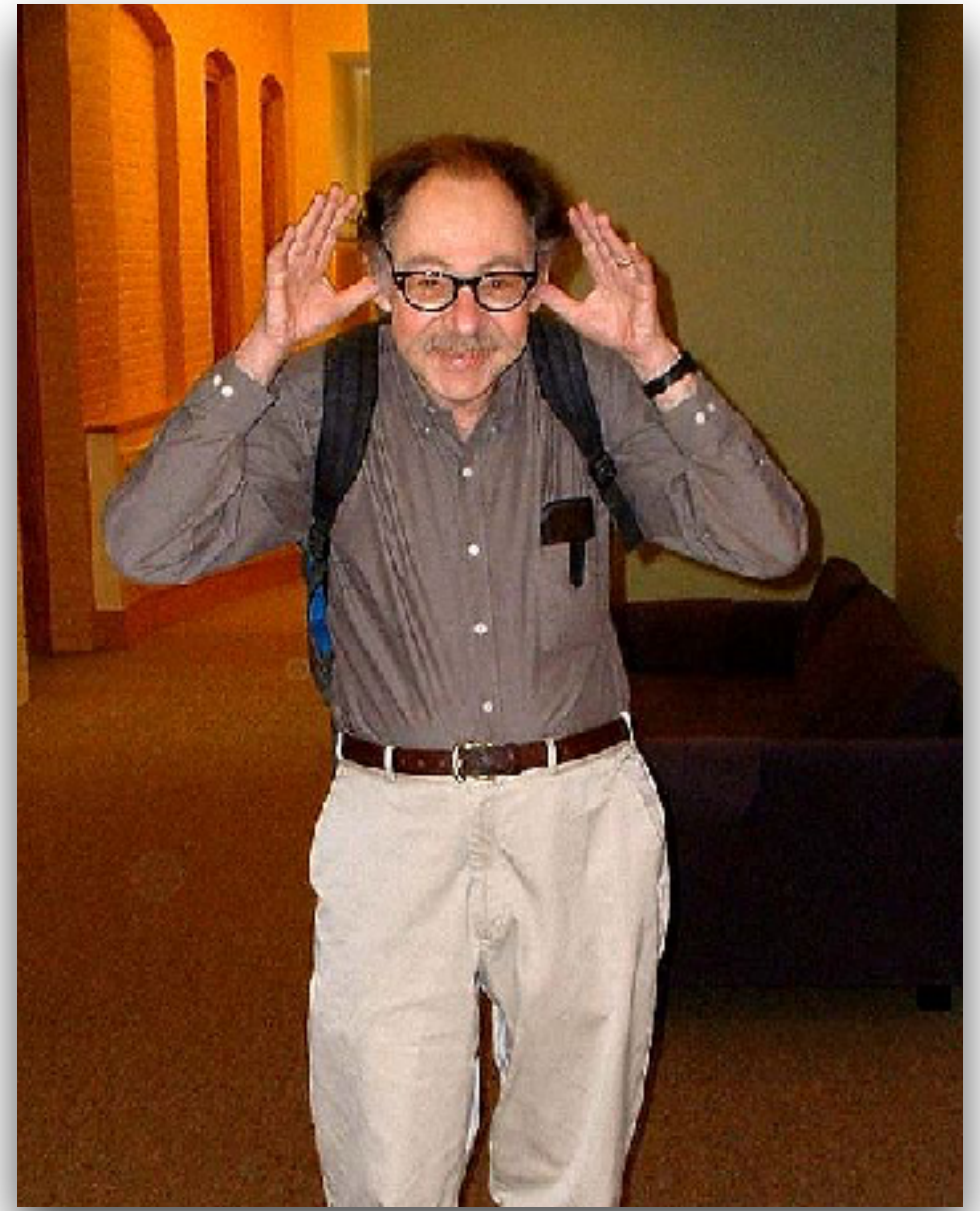


https://repl.it/@enaard/Python-3

Scan me

Complete code examples available from this lesson

"The career of a young theoretical physicist consists of treating the harmonic oscillator in ever-increasing levels of abstraction"



Sidney Coleman
1937-2007

a basic harmonic oscillator…

mass and spring system

Restoring force -

spring tension

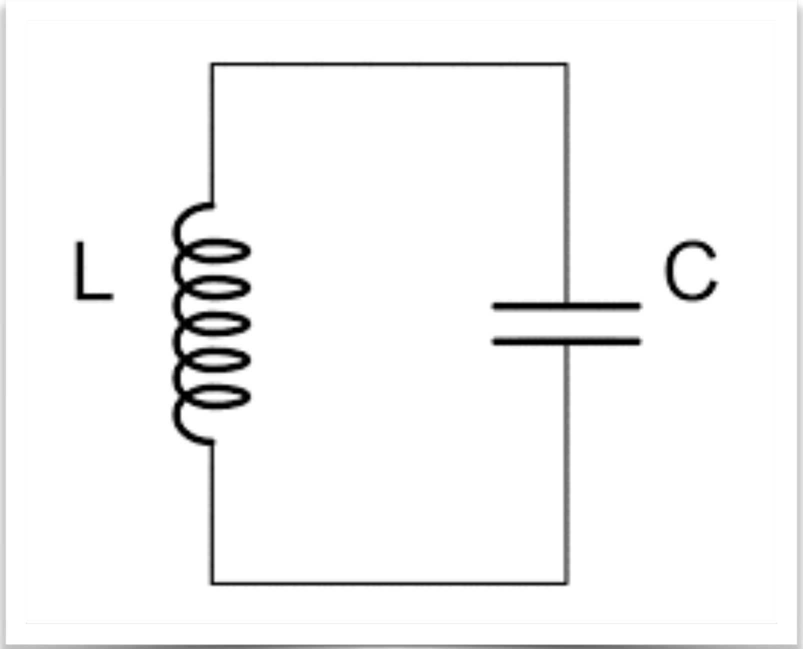
Restoring force $\propto$ (-) displacement

$$F = -kx$$

harmonic motion…

pendulum clock

stringed instrument



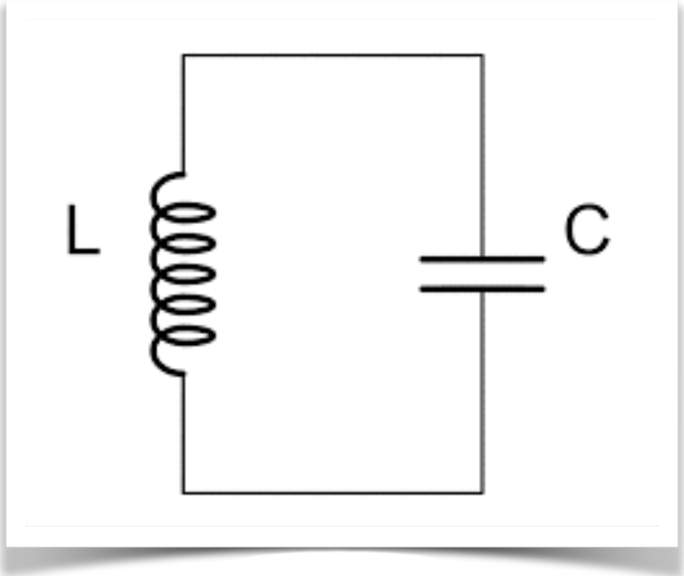
buoy

electronic circuits

…many examples!

What is it?

Harmonic motion occurs due to a ***restoring force***



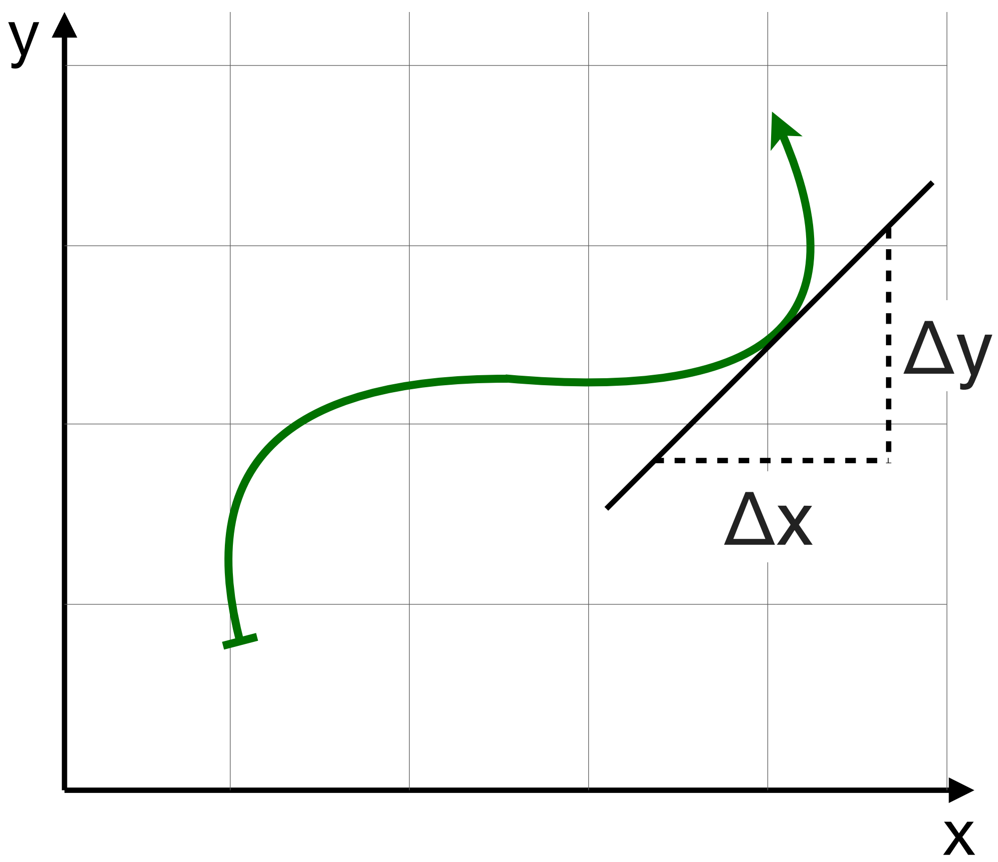
Restoring force - ***gravity***

Restoring force - ***tension***

Restoring force - ***electric current***

what does harmonic motion look like?

we will use the concept of a derivative - used to study quantities that are changing



the gradient (slope) of the curve is

$$\frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$$

we will write this as

$$\frac{dy}{dx} \simeq \frac{\Delta y}{\Delta x}$$

a little maths…

Newton's second law

$$F = \frac{d}{dt} p(t)$$

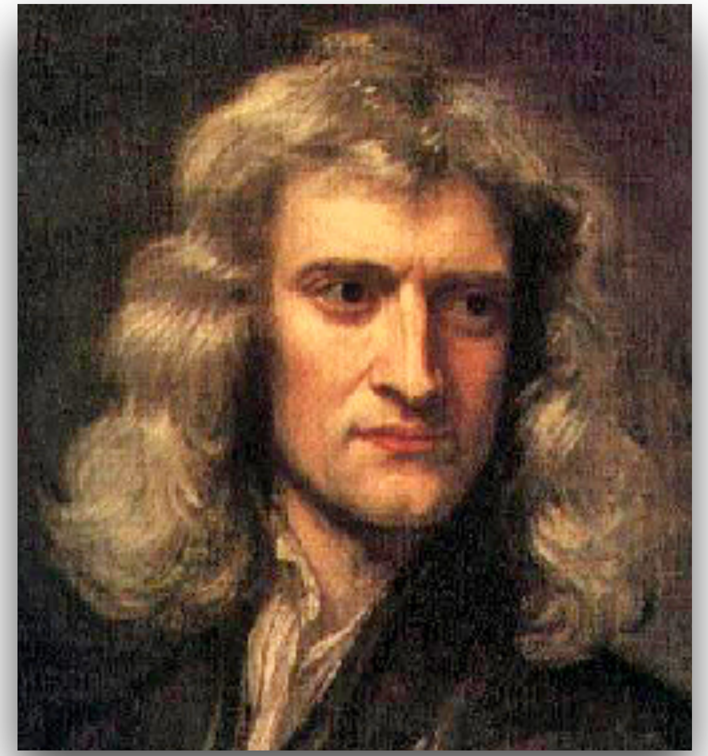
- Force = *rate of change* of momentum

$$p(t) = m \frac{d}{dt} x(t)$$

- Momentum = mass x velocity

$$F = m \frac{d^2}{dt^2} x(t)$$

- **Force = mass x acceleration**



Issac Newton
1642-1726

a little more maths…

Hook’s displacement Law

$$F_{\text{restore}} = -kx$$

- Restoring force described by *Hook’s law*

$$F_{\text{Newton}} = F_{\text{restore}}$$

- Set force is equal to restoring force

$$m\frac{d^2}{dt^2}x(t) = -kx(t)$$

- Differential equation for *mass spring system*

Robert Hook
1635-1703

final bit of maths…

Harmonic oscillator solution

$$m\frac{d^2}{dt^2}x(t) + kx(t) = 0$$

- *x(t)* is a *periodic* function -

$$x(t + 2\pi n) = x(t)$$

- let’s try a trigonometric function

$$x(t) = A\cos(\omega t + \phi)$$

- substitute into differential equation

$$\omega = \sqrt{k/m}$$

- *ω* defines *natural frequency* of oscillation

Scripting time!

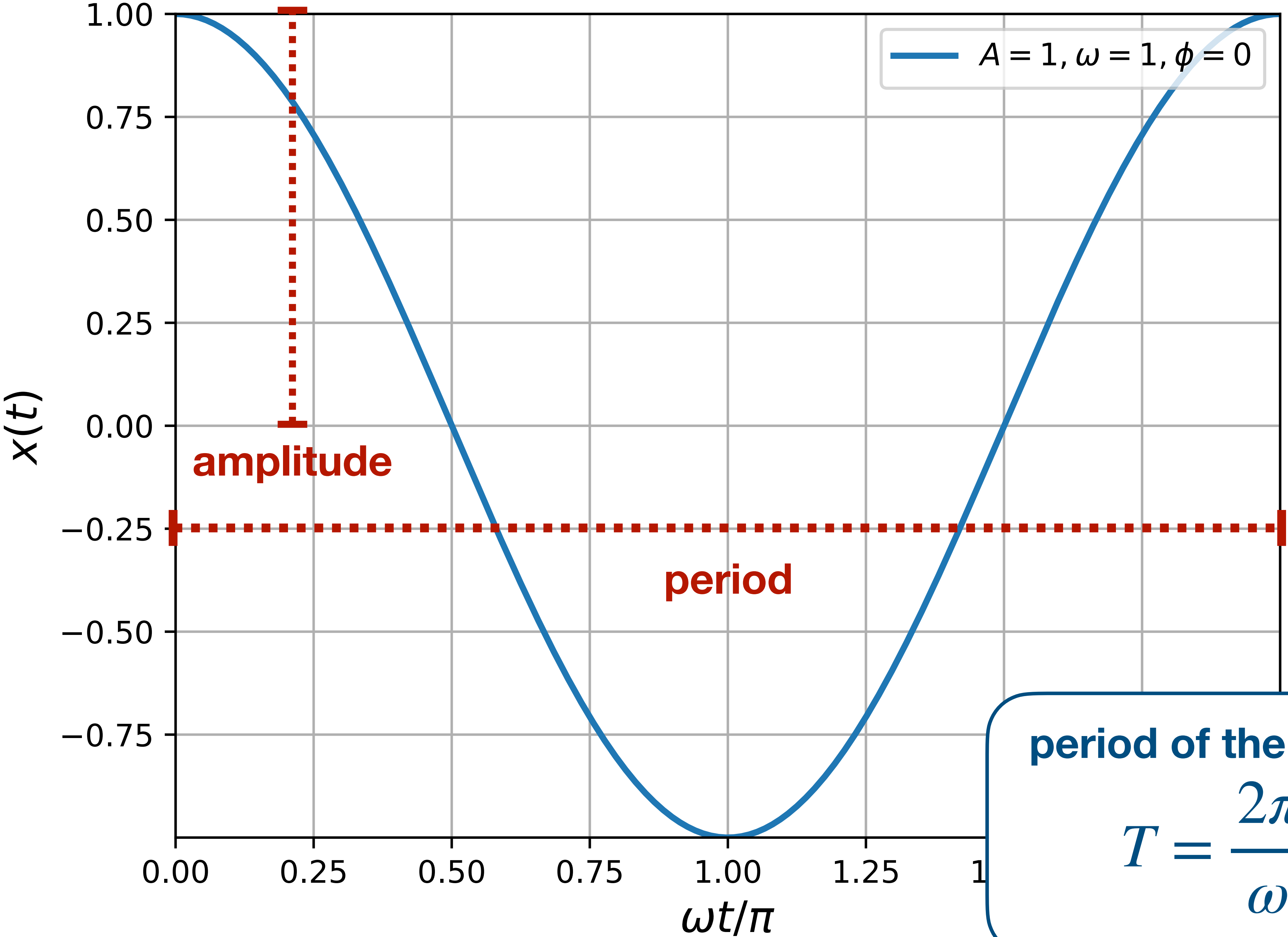
```
import numpy as np
import matplotlib.pyplot as plt
#
phi = 0
w = 1
t = np.linspace(0,2*np.pi,1e3)
x = np.cos(w*t+phi)
#
my_fig = plt.figure()
plt.plot(t,x)
plt.xlabel('wt', fontsize=14)
plt.ylabel('x(t)', fontsize=14)
my_fig.show()
#
```

python libraries we wish to use

define parameters and variables we wish to plot

define the figure and plot the variables

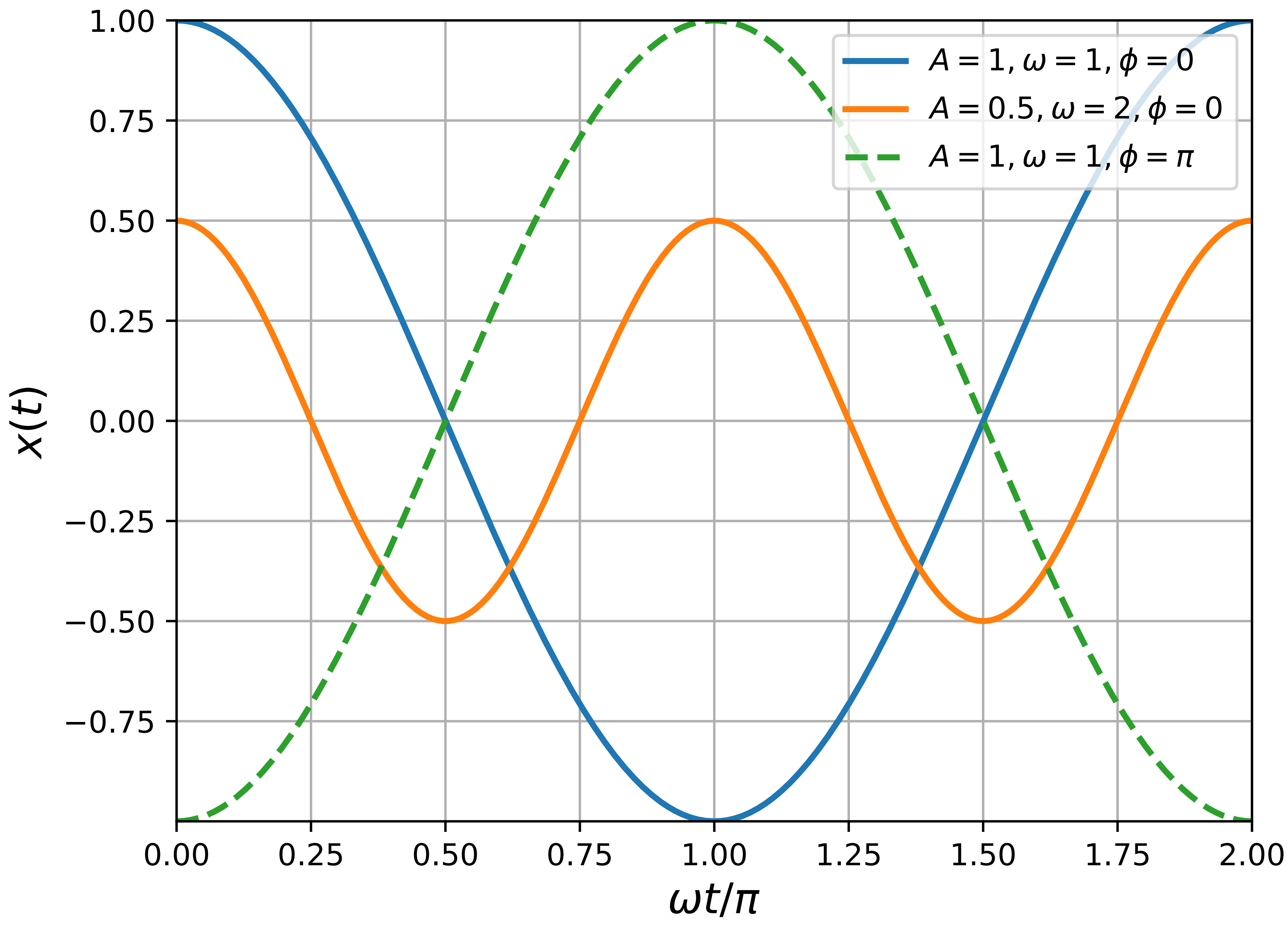
What do the solutions look like?



period of the wave

$$T = \frac{2\pi}{\omega}$$

What do the solutions look like?



What if we don't know the solution?

most models do not have analytical solutions!

use *quadrature* (numerical integration)

Finite difference method

- Approximate derivative as

$$\frac{dx}{dt} \approx \frac{x(t+\Delta) - x(t)}{\Delta}$$

- Known as *discretization*
- Very useful for solving simple and more complex models

Harmonic oscillator example

$$m\frac{d^2}{dt^2}x(t) + kx(t) = 0$$

- ***We can solve using quadrature!***

- First, let's separate into two coupled equations:

$$\frac{d}{dt}p(t)+\omega^2 x(t) = 0$$

$$p(t) = \frac{d}{dt}x(t)$$

- Now we can solve numerically using *Python*

- Requires *two* initial conditions

harmonic oscillator example

$$m\frac{d^2}{dt^2}x(t) + kx(t) = 0$$

- ***Let's discretize our model…***

- For some function *f(t)* the derivative can be approximated as

$$\frac{df}{dt} \simeq \frac{f(t+\Delta) - f(t)}{\Delta}$$
$$= \frac{1}{\Delta}\left(f_{i+1} - f_i\right)$$

- Finite difference method - used widely in STEM

$$p_{t+1} = p_t - \Delta\omega^2 x_t$$
$$x_{t+1} = \Delta p_t + x_t$$

a note on iteration

understanding discretization

we could write out each line…

```
f[1] = …
f[2] = …
f[3] = …
```

incredibly inefficient!

- p and x can be represented by a (finite) list of numbers
- the numbers in the list are equally spaced (by Δ)
- we can use iteration to update the list

```
for i = 0 : N
  #
    f[i] = i;
    print(f[i]);
  #
```

`for` iteration scheme *very* useful!

Scripting time! (#2)

```
import numpy as np
import matplotlib.pyplot as plt
#
dt = 1e-3
w = 1
T = 2*np.pi
Nt = round(T/dt)
tsc = np.linspace(0,T,Nt)
#
x = np.zeros(Nt);
p = np.zeros(Nt);
x[0] = 1
p[0] = 0
#
```

initial conditions, parameters for numerical integration

define position and momentum vectors, apply initial conditions

Scripting time! (#2)

for loop *iterates* through both vectors

```
for jj in range(1,Nt-1):
    #
    x[jj+1] = dt * p[jj] + x[jj]
    p[jj+1] = p[jj] - dt * w**2 * x[jj]
    #
#
```

at each step of the loop, the *x(t)* and *p(t)* vectors are updated

$$x_{t+1} = \Delta p_t + x_t$$

$$p_{t+1} = p_t - \Delta \omega^2 x_t$$

Scripting time! (#2)

Let's plot the data, with legend and axis labels

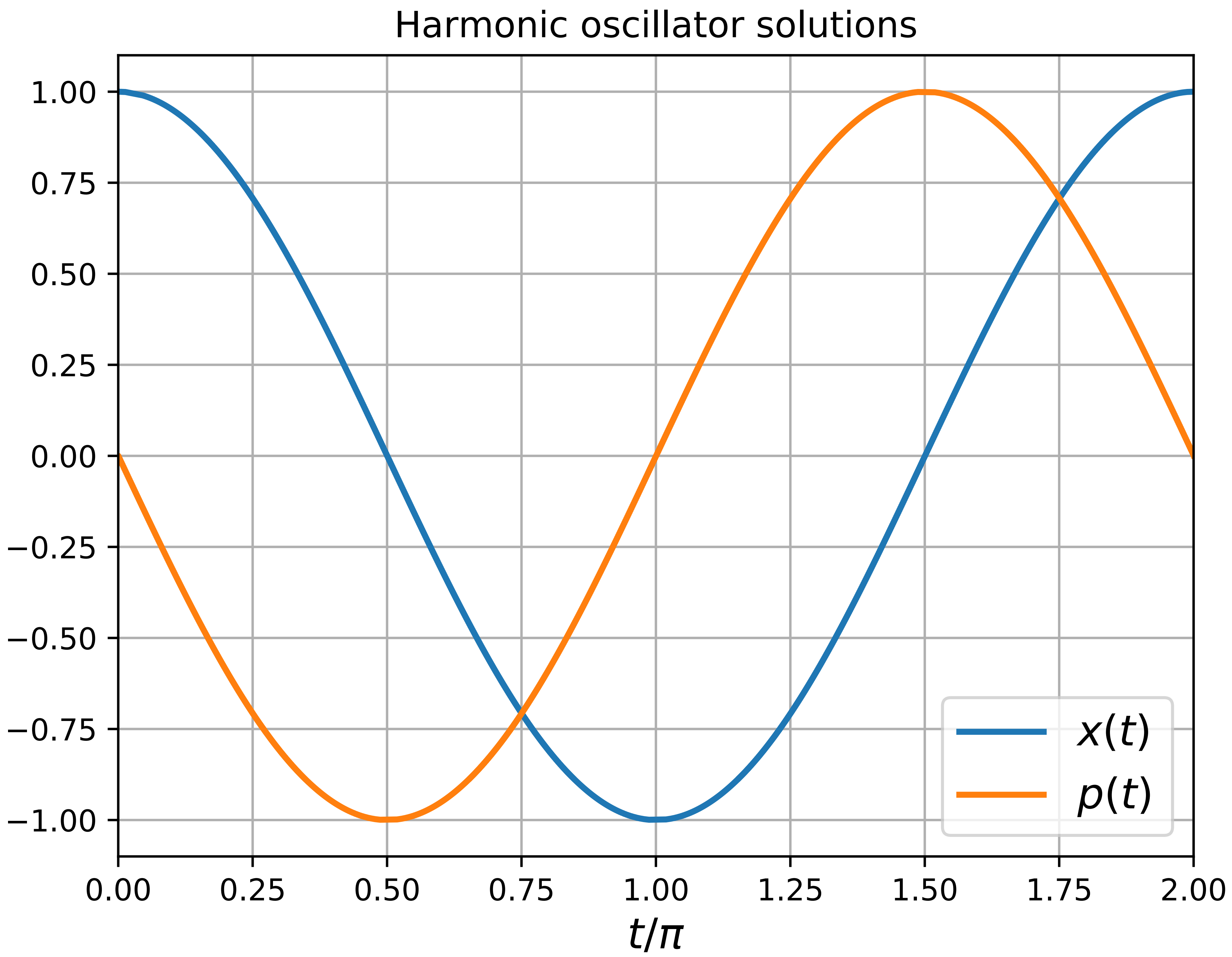
```
#
f = plt.figure()
data = plt.plot(tsc/np.pi,x,tsc/np.pi,p,linewidth='2')
plt.legend(data, ('$x(t)$', '$p(t)$'), fontsize='14')
plt.title('Harmonic oscillator solutions')
#
```

label the x-axis, make the axis fit the data and show the grid

create a new figure plotting *x(t)* and *p(t)* vs time

```
#
plt.xlabel(r'$t/\pi$', fontsize='14')
plt.autoscale(enable=True, axis='x', tight=True)
plt.grid()
f.show()
#
```

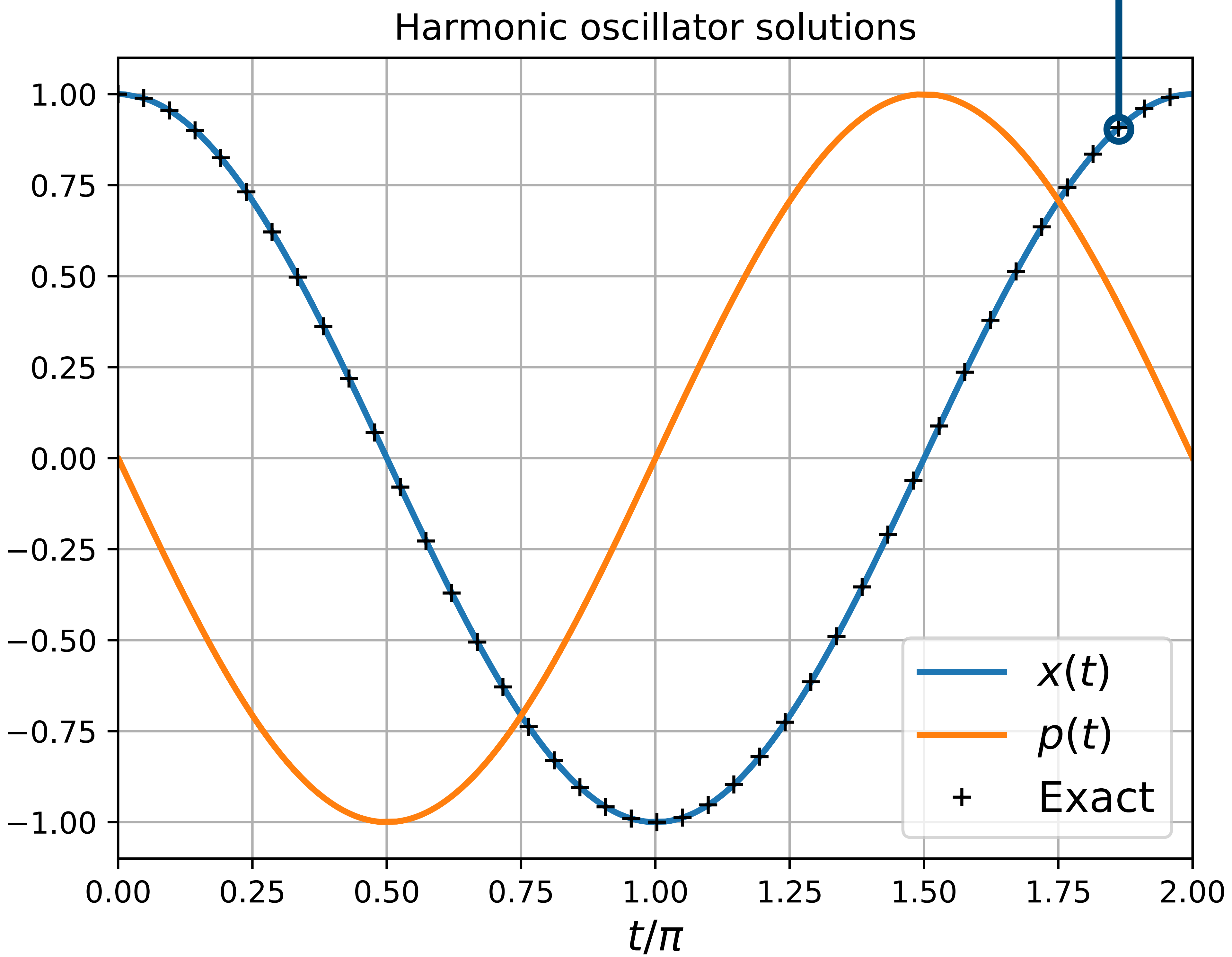
The result…


Harmonic oscillator solutions
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
t/π
$x(t)$
$p(t)$

The result…

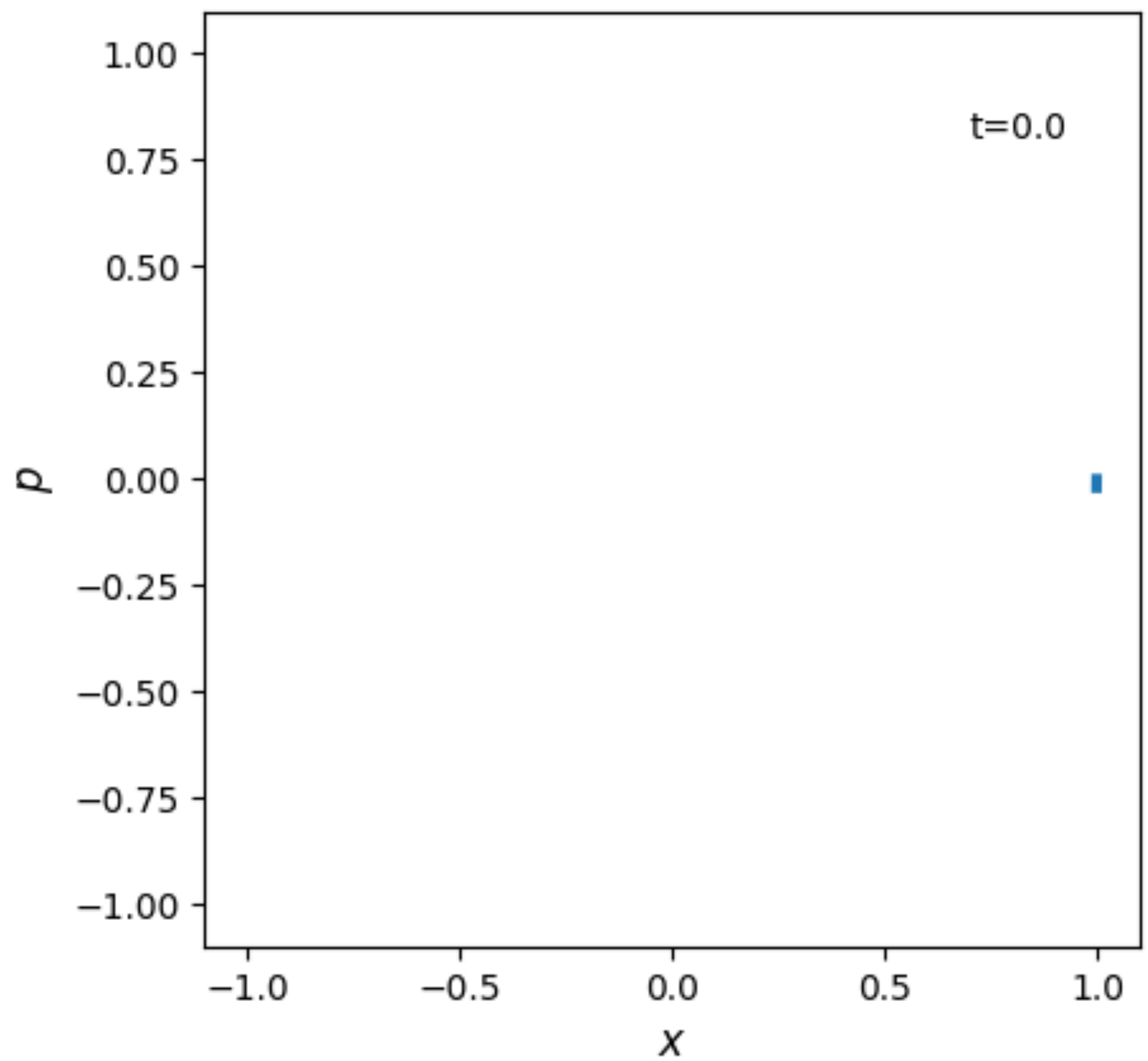
What if we compare with the exact solution?

$$x(t) = A\cos(\omega t + \phi)$$



a closer look…

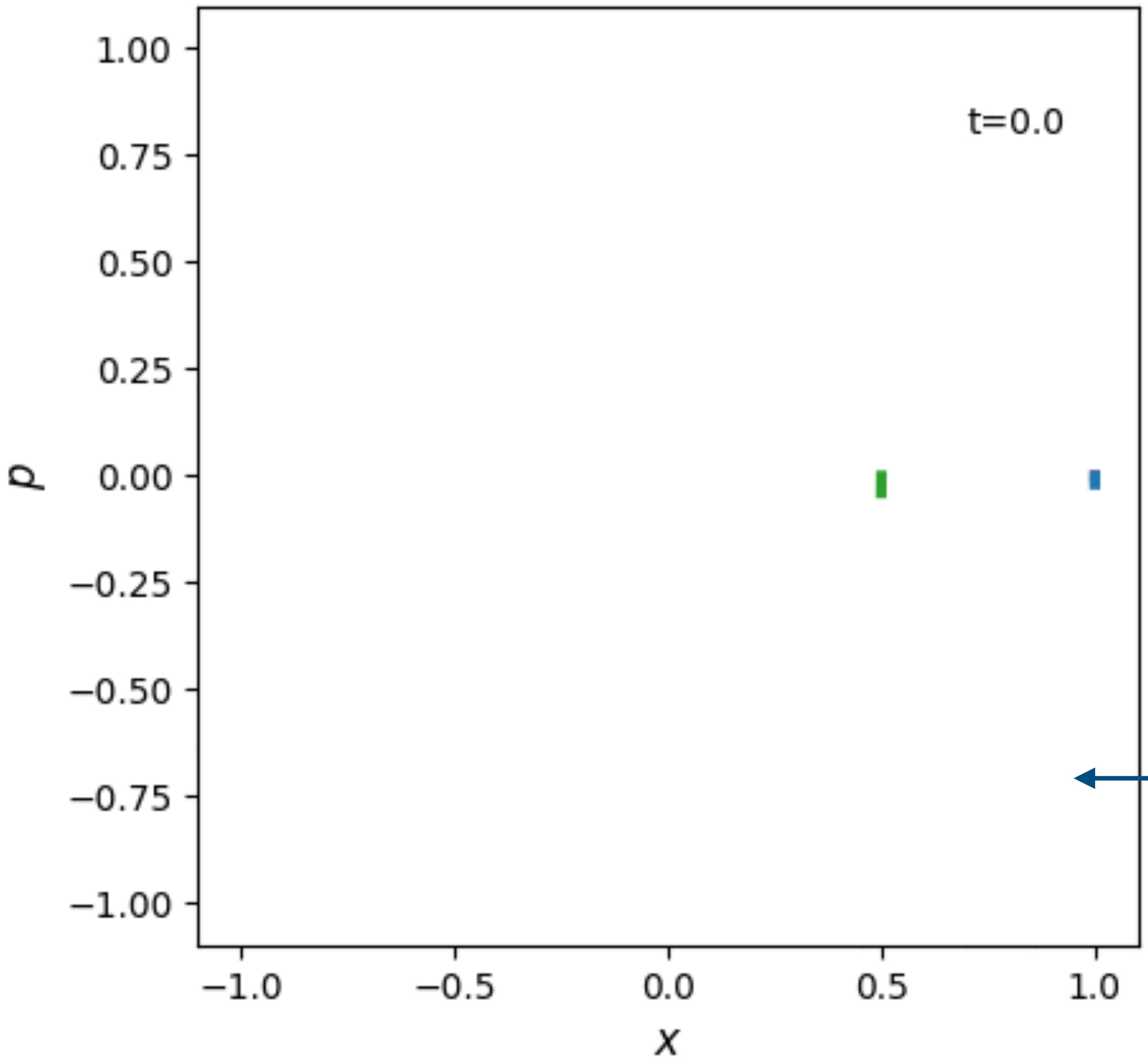
What if we plot position against momentum?



x vs. p ***phase plot*** very important concept!

a closer look…

What if we plot position against momentum?



	data key
(blue)	$x_0=1 \quad w=1$
(red)	$x_0=1 \quad w=½$
(green)	$x_0=½ \quad w=2$

curves form *ellipses* in the phase plot

a more interesting example

Q: *What real effect is missing from the harmonic oscillator model?*

A: The simple harmonic oscillator does not include *damping*!

- damping opposes the motion of the particle

$$F_{\text{damp}} = \gamma \frac{dx}{dt}$$

- damping force depends on the *velocity* of the particle

$$F_{\text{all}} = F_{\text{damp}} + F_{\text{restore}}$$

- new equation of motion for particle-

$$m\frac{d^2x}{xt^2} + \gamma\frac{dx}{dt} + kx = 0$$

Friction - some examples

between solids and gases

between solid objects

between atoms in a fluid

including the term for friction

equation describing *damped* harmonic oscillator	$$m\frac{d^2x}{dt^2}+\gamma\frac{dx}{dt}+kx=0$$
separate into pair of coupled equations	$$\frac{d}{dt}p(t)+\frac{\gamma}{m}p(t)+\omega^2x(t)=0$$ $$p(t)=\frac{d}{dt}x(t)$$
discrete coupled equations for particles position and momentum	$$p_{t+1}=p_t-\Delta\left(\frac{\gamma}{m}p_t+\omega^2x_t\right)$$ $$x_{t+1}=\Delta p_t+x_t$$

let's modify our python script

```python
#
g = 1
#
```

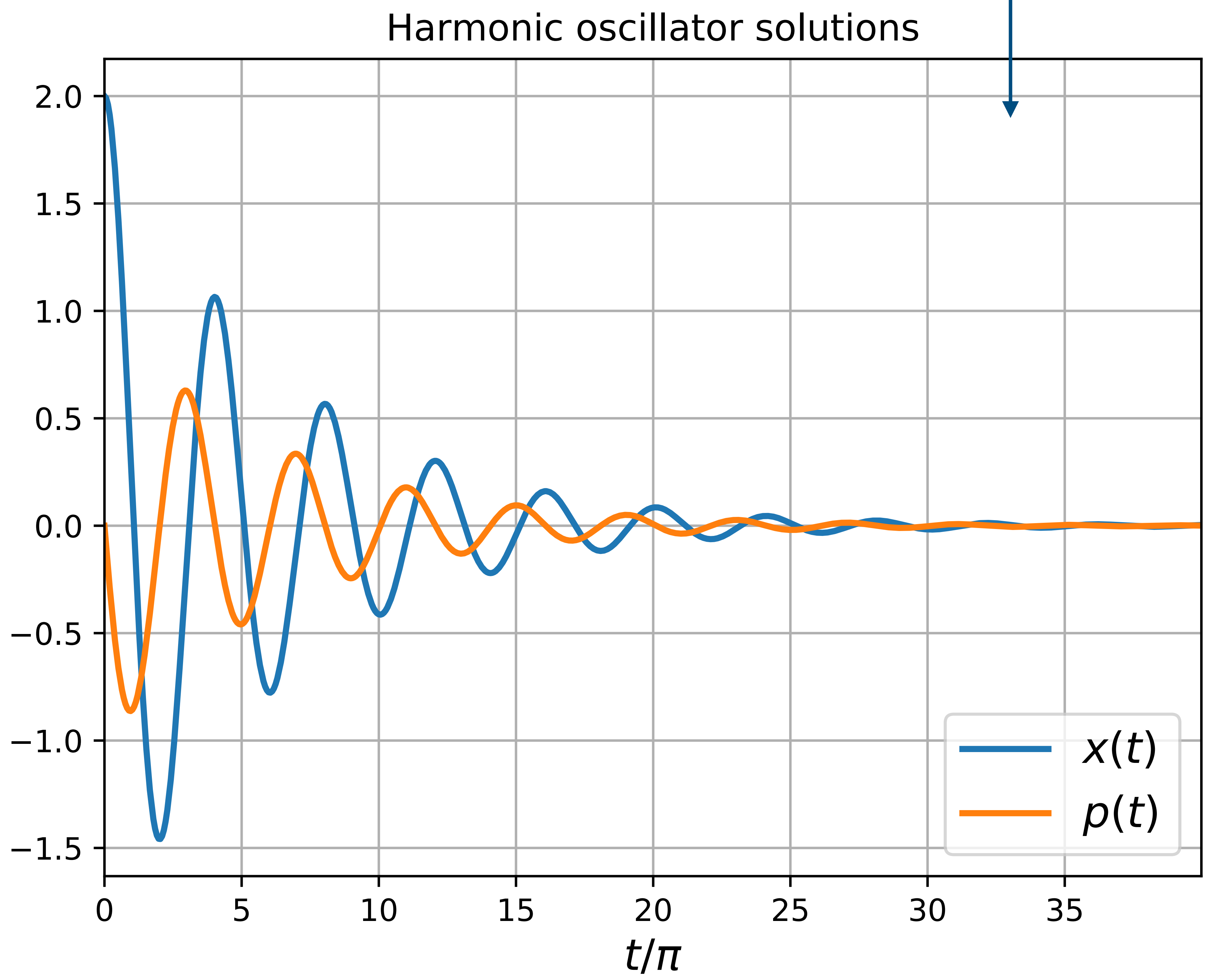
new parameter - damping strength g

```python
#
for jj in range(1,Nt-1):
    #
    x[jj+1] = x[jj] + dt*p[jj]
    p[jj+1] = p[jj] - dt*(g*p[jj] + w**2 * x[jj])
    #
#
```

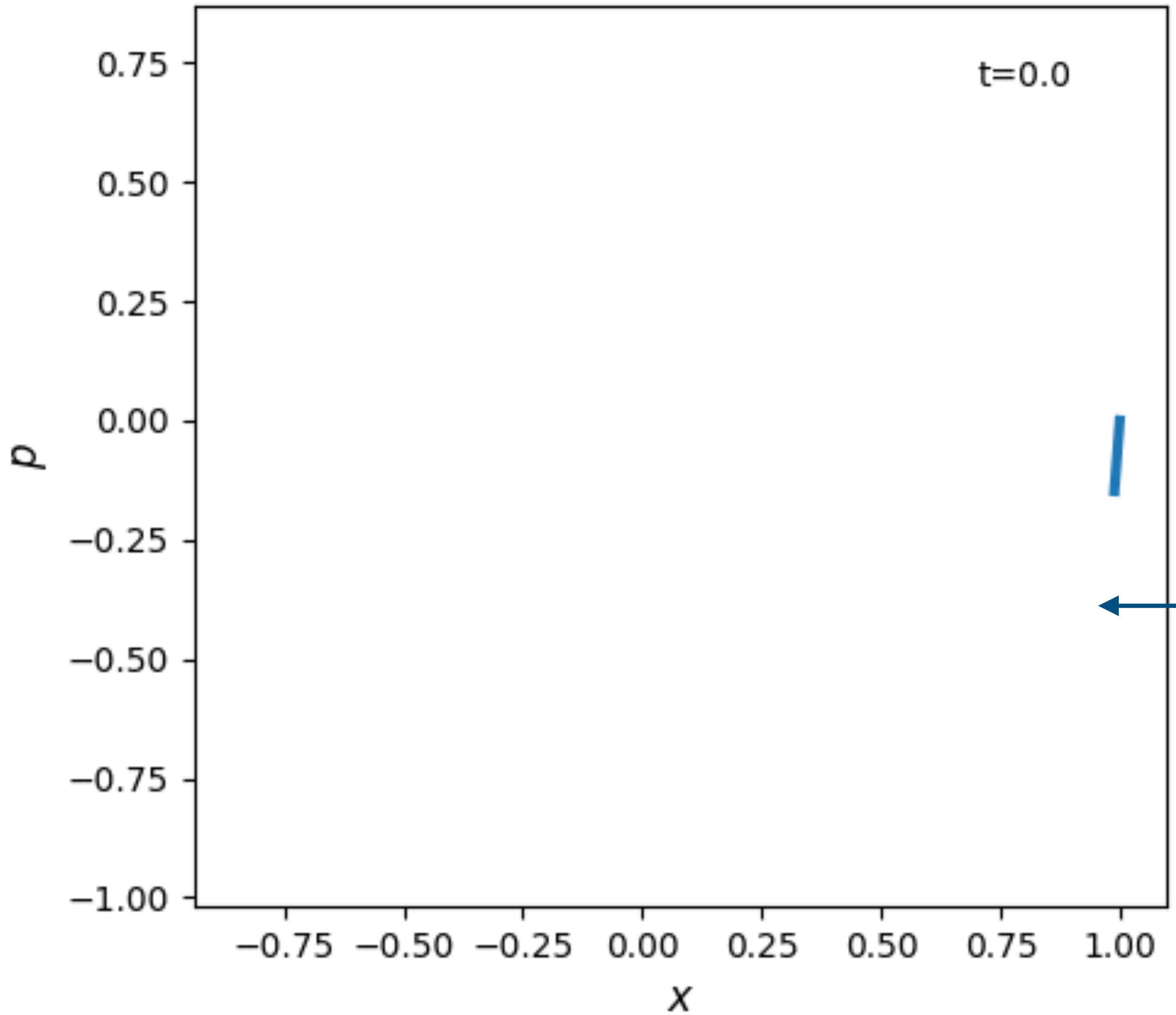
damping term only modifies second equation

The result…

Motion is damped, restoring force is suppressed.



phase plot

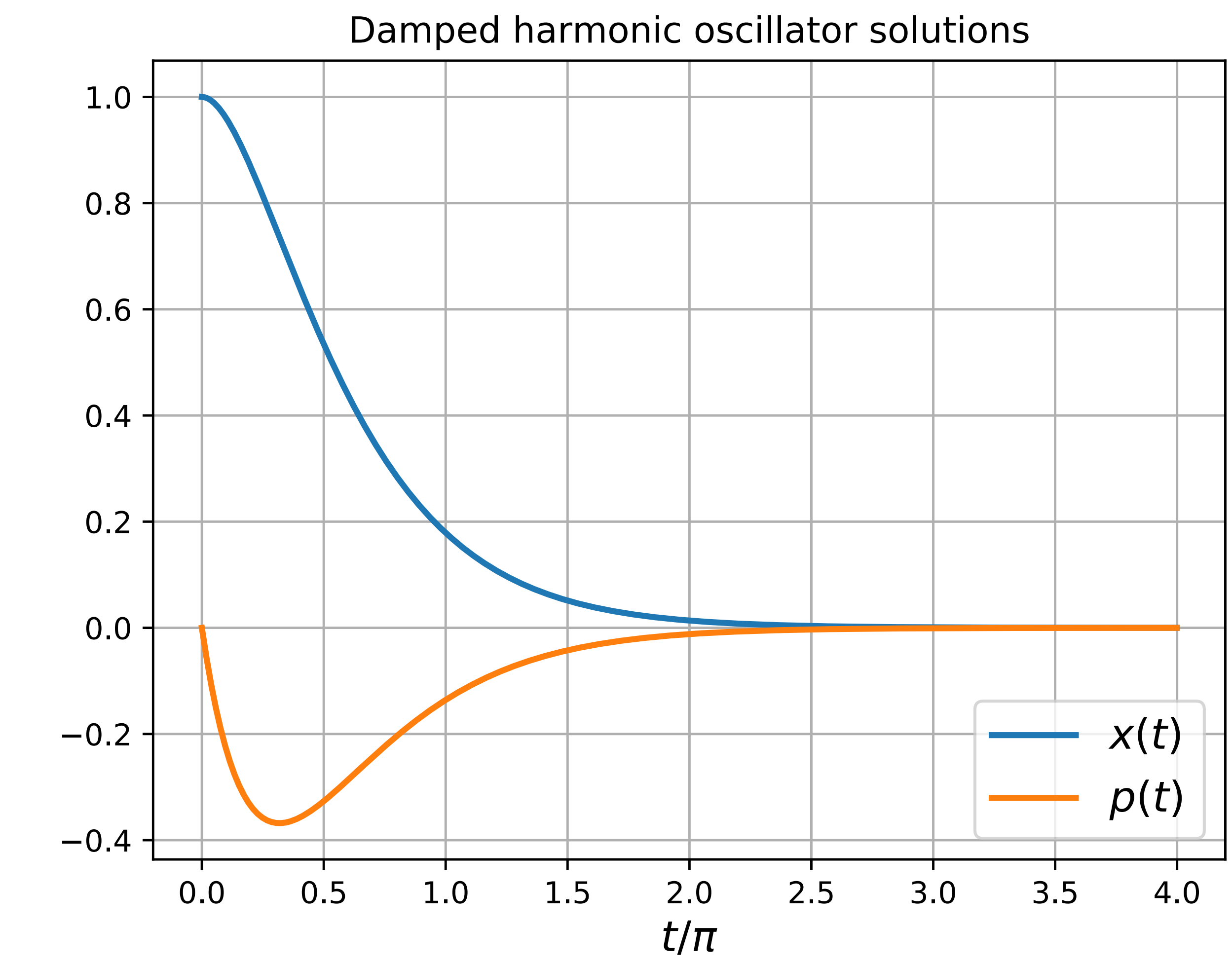


phase plot forms *spiral* due to damping

eventually, the particle is *localised*

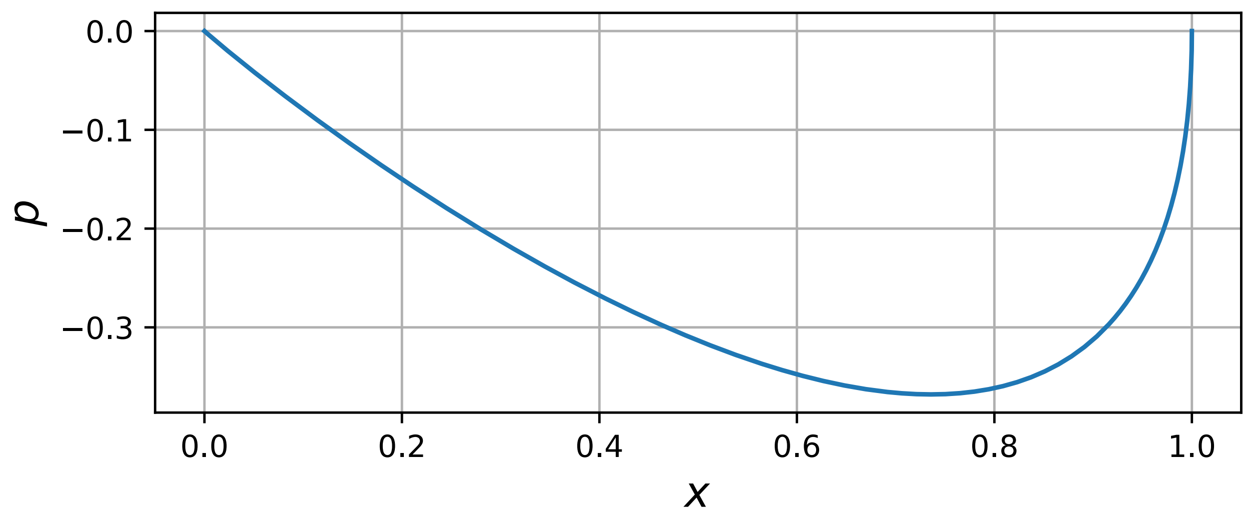
strong damping

What happens when there is strong damping?



parameters

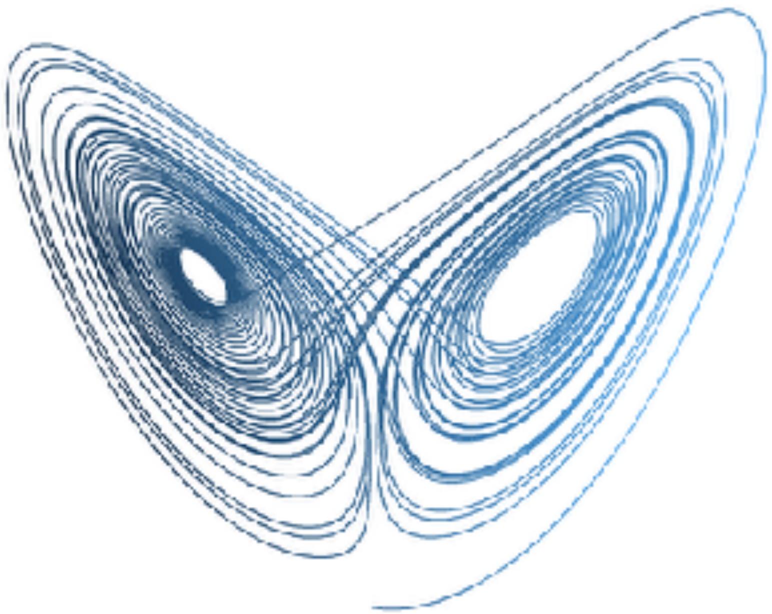
$x_0 = 1$
$w = 1$
$g = 2$



Phase plot shows *fast* localization

outlook

where can we go from here?



chaotic systems
strange attractors

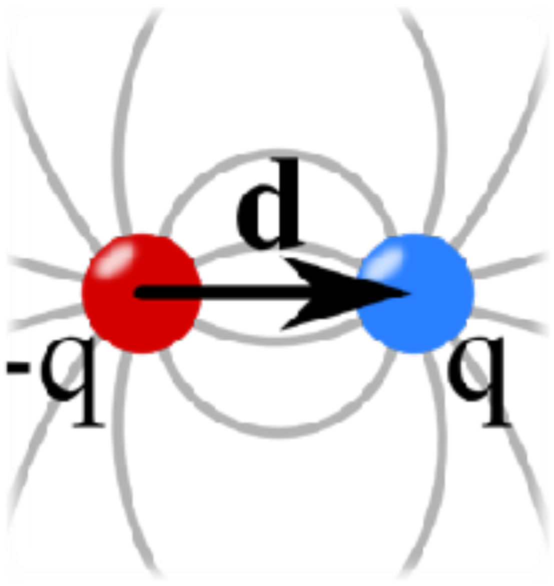
Tacoma narrows bridge
(c.1940)

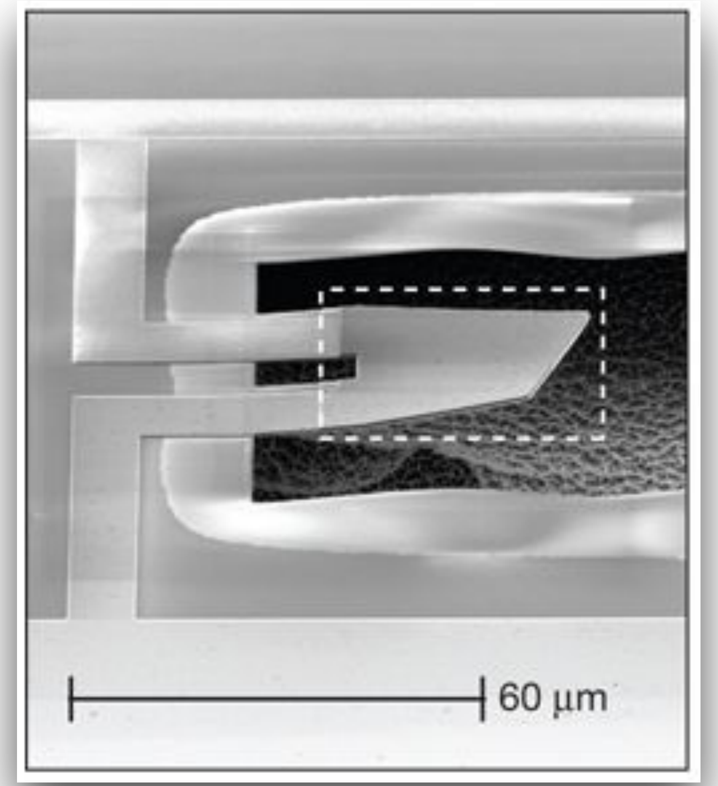
forced oscillators
e.g. swing

stability important for
designing buildings

outlook

where can we go from here?





dipole force

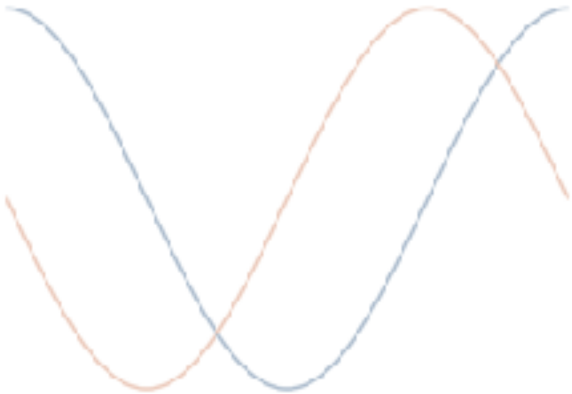
quantum devices

climate physics

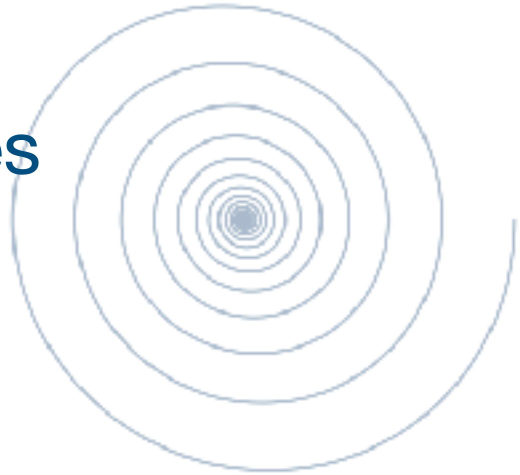
SHO building block for understanding many more complicated systems

summary and learning outcomes

- simple harmonic oscillator describes *periodic motion*



- discrete model can be solved *numerically*
- python implementation explained
- damped harmonic motion explored
- phase plots introduced with different examples



Scan me

Complete code examples available from this lesson