Numerical computation on equation of harmonic oscillator

Huang cheng

Abstract:

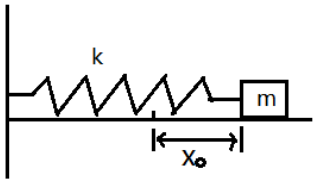
An efficient method of numerical computation on harmonic oscillator is investigated by using Euler method as well as Talor expansion.The effect of the values of time step and the orders of approximation are discussed.The results agree well with mathematical analysis.The author also draws a 3D figure.

Background:

The equation of harmonic oscillator is of second order.

Text:

The equation of harmonic oscillator is $\frac{d^2x}{dt^2}=-\frac{k}{m}x$,where m is the mass of oscillator and k is Hooker coefficient,as shown in the picture on the right.



We take m as $0.1kg$, k as $0.9N/m$. Thus $\frac{k}{m}=9$. The initial conditions is settled to be $t=0, v(0)=0, x(0)=2$.

So,the analytical solution is $x=2\sin\left(3t+\frac{\pi}{2}\right), v=6\cos\left(3t+\frac{\pi}{2}\right)$.

Next, we would resolve the equation by using numerical computation.

$\frac{d^2x}{dt^2} = -\frac{k}{m}x$ is rewritten as $\begin{cases} \frac{dv}{dt} = -9x \\ v = \frac{dx}{dt} \end{cases}$.

As a result,Euler method or Talor expansion is available for x and v ,respectly.

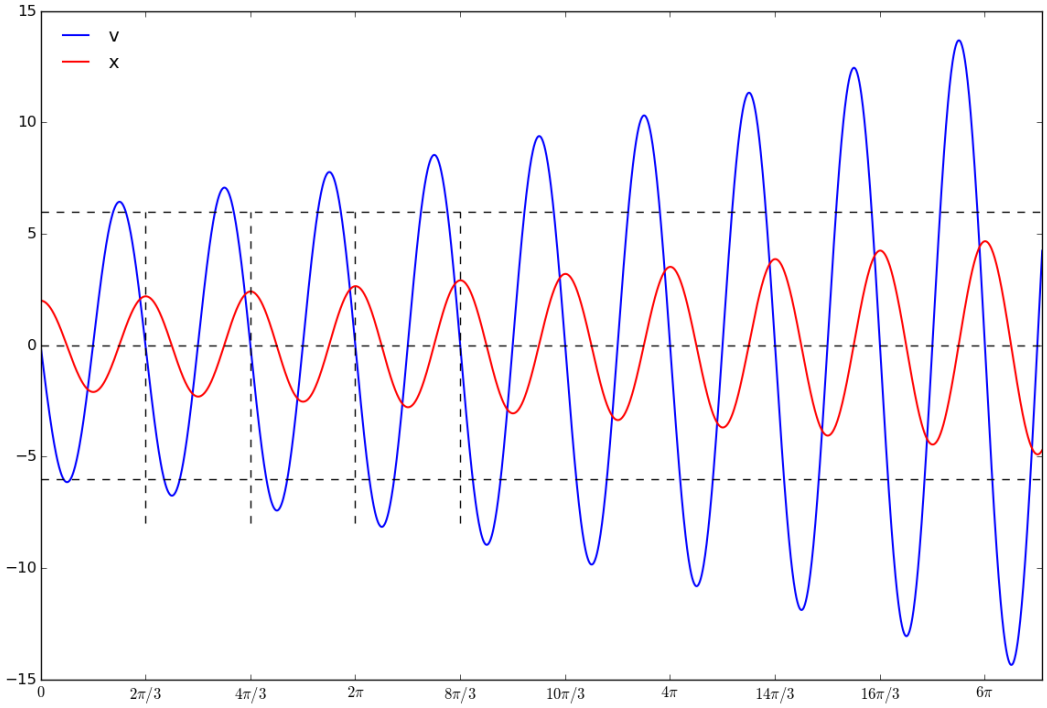
To begin with,we use Euler method.

That is
$$v(t+dt) \approx v(t) + \frac{dv}{dt} \cdot dt = v(t) - 9xdt$$
$$x(t+dt) \approx x(t) + \frac{dx}{dt} \cdot dt = x(t) + vdt$$

The main program portion is(the complete code is available in another file):

```
v=[ ]

x=[ ]

t=[ ]

v.append(0)

x.append(2)

t.append(0)

total_time=20

dt=0.01

N=total_time/dt

for i in range(int(N)):

    v.append(v[i]-9*x[i]*dt)

    x.append(x[i]+v[i]*dt)

    t.append(dt*(i+1))
```

If $dt = 0.01$,the result is



which is not satisfactory.

Reasons that account for this terrible result are:

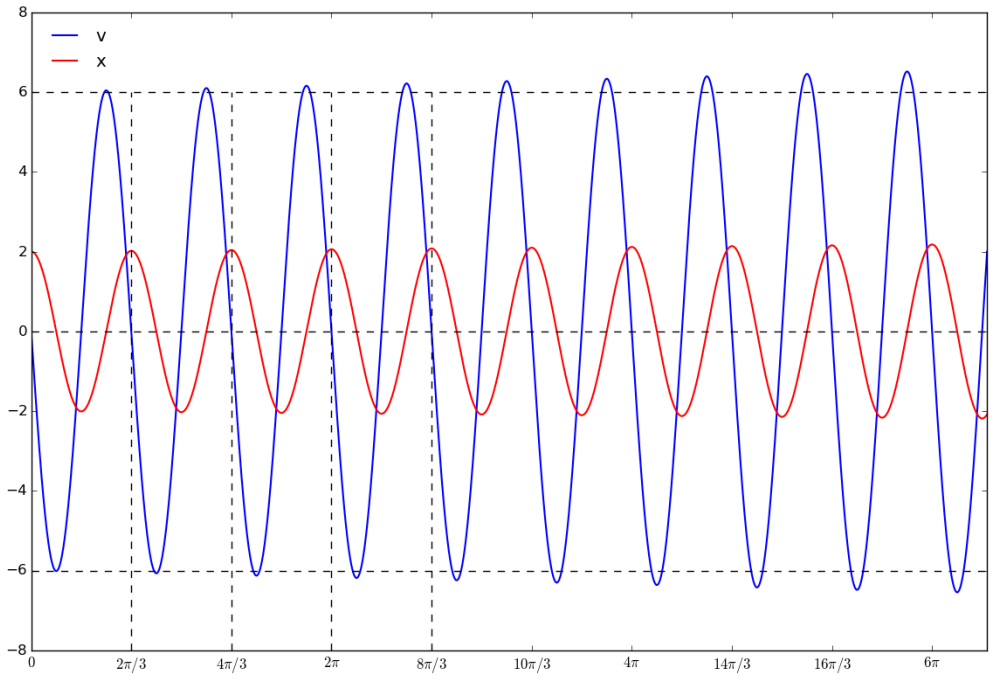
①the value of time step is still too large,

②or the Euler approximation is not good enough.

It is suggested that we decrease the value of dt or use higher order approximation or both.

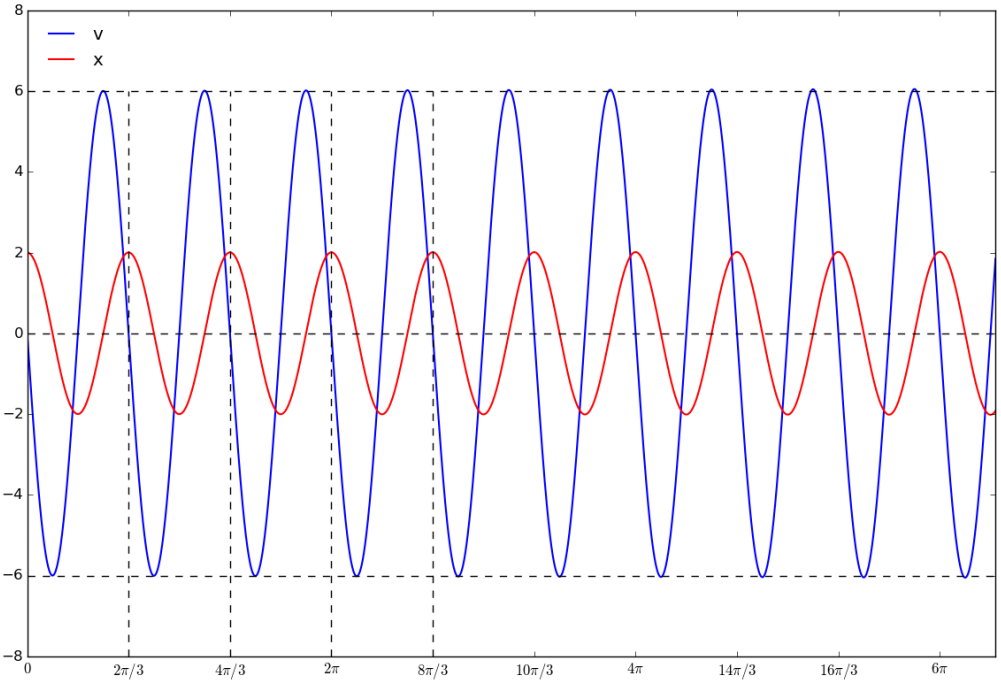
Here we decrease dt in the first place.

If $dt = 0.001$,corresponding result is



which is not so good either.

If $dt = 0.0001$,corresponding result is



which is pretty good.And its consistent with analytical solution.

In the following,we use higher approximation due to Talor expansion.And set $dt = 0.01$.

Second order approximation:

$$\frac{d^2v}{dt^2} = \frac{d}{dt}\frac{dv}{dt} = \frac{d}{dt}(-9x) = -9v$$

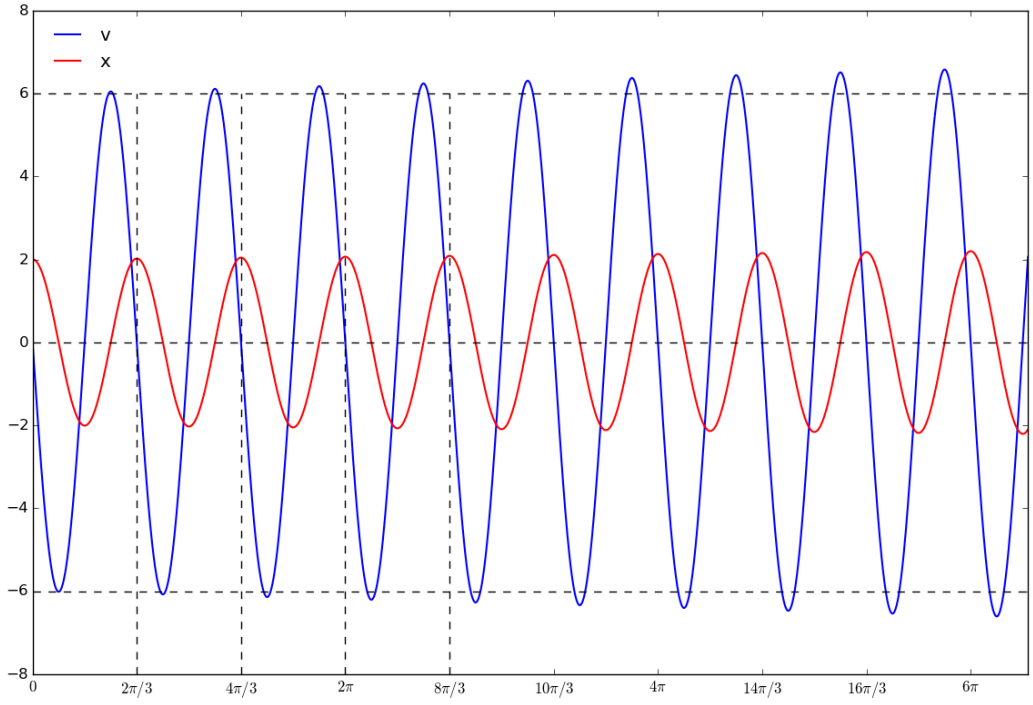
$$v(t+dt) \approx v(t) + \frac{dv}{dt}\cdot dt + \frac{1}{2}\frac{d^2v}{dt^2}\cdot(dt)^2 = v(t) - 9xdt - \frac{9}{2}v\cdot(dt)^2$$

$$x(t+dt) \approx x(t) + \frac{dx}{dt}\cdot dt + \frac{1}{2}\frac{d^2x}{dt^2}\cdot(dt)^2 = x(t) + vdt - \frac{9}{2}x\cdot(dt)^2$$

Main code is changed to

```
for i in range(int(N)):
    v.append(v[i]-9*x[i]*dt-9/2*v[i]*dt**2)
    x.append(x[i]+v[i]*dt-9/2*x[i]*dt**2)
    t.append(dt*(i+1))
```

Corresponding result is



It looks much better than that of without second correction.

Further,the third order approximation:

$$\frac{d^3v}{dt^3}=81x,\frac{d^3x}{dt^3}=-9v$$

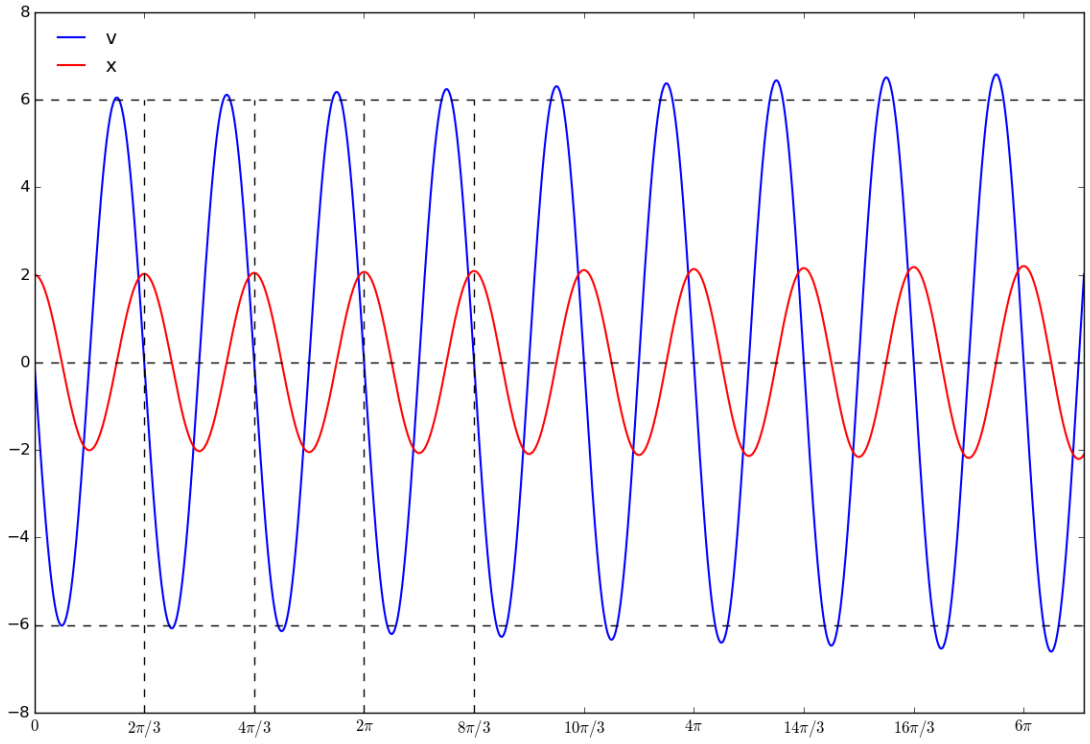
$$v(t+dt)\approx v(t)+\frac{dv}{dt}\cdot dt+\frac{1}{2}\frac{d^2v}{dt^2}\cdot(dt)^2+\frac{1}{6}\frac{d^3v}{dt^3}\cdot(dt)^3=v(t)-9xdt-\frac{9}{2}v\cdot(dt)^2+\frac{27}{2}x\cdot(dt)^3$$

$$x(t+dt)\approx x(t)+\frac{dx}{dt}\cdot dt+\frac{1}{2}\frac{d^2x}{dt^2}\cdot(dt)^2+\frac{1}{6}\frac{d^3x}{dt^3}\cdot(dt)^3=x(t)+vdt-\frac{9}{2}x\cdot(dt)^2-\frac{3}{2}v\cdot(dt)^3$$

Main code is changed to

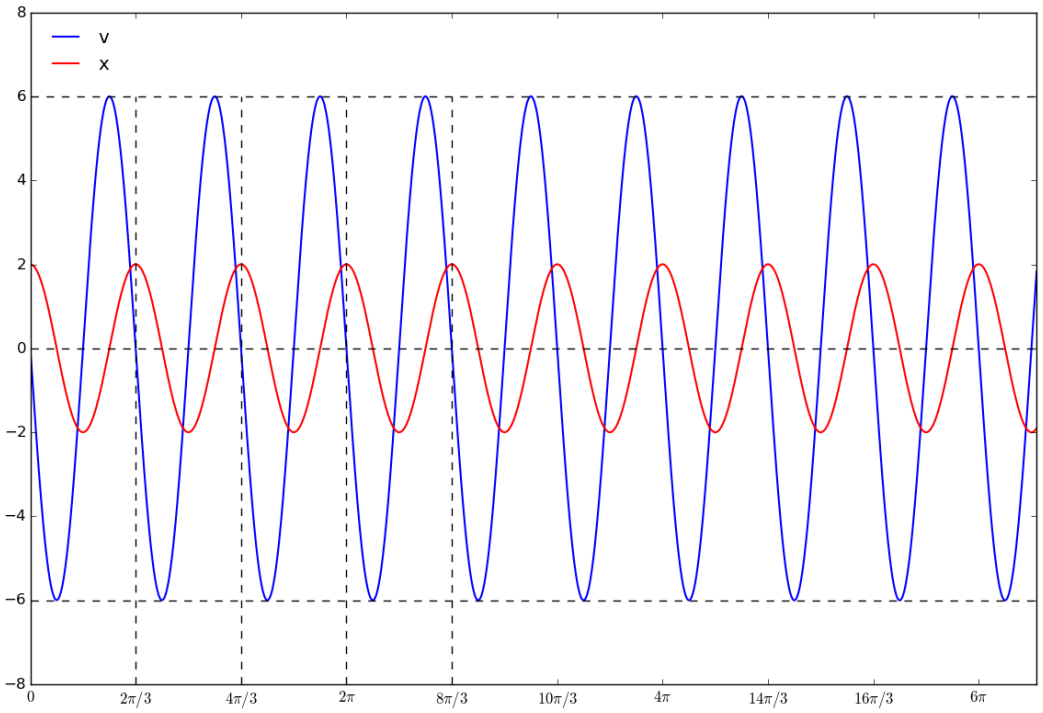
```
for i in range(int(N)):
    v.append(v[i]-9*x[i]*dt-9/2*v[i]*dt**2+27/2*x[i]*dt**3)
    x.append(x[i]+v[i]*dt-9/2*x[i]*dt**2-3/2*v[i]*dt**3)
    t.append(dt*(i+1))
```

Corresponding result is



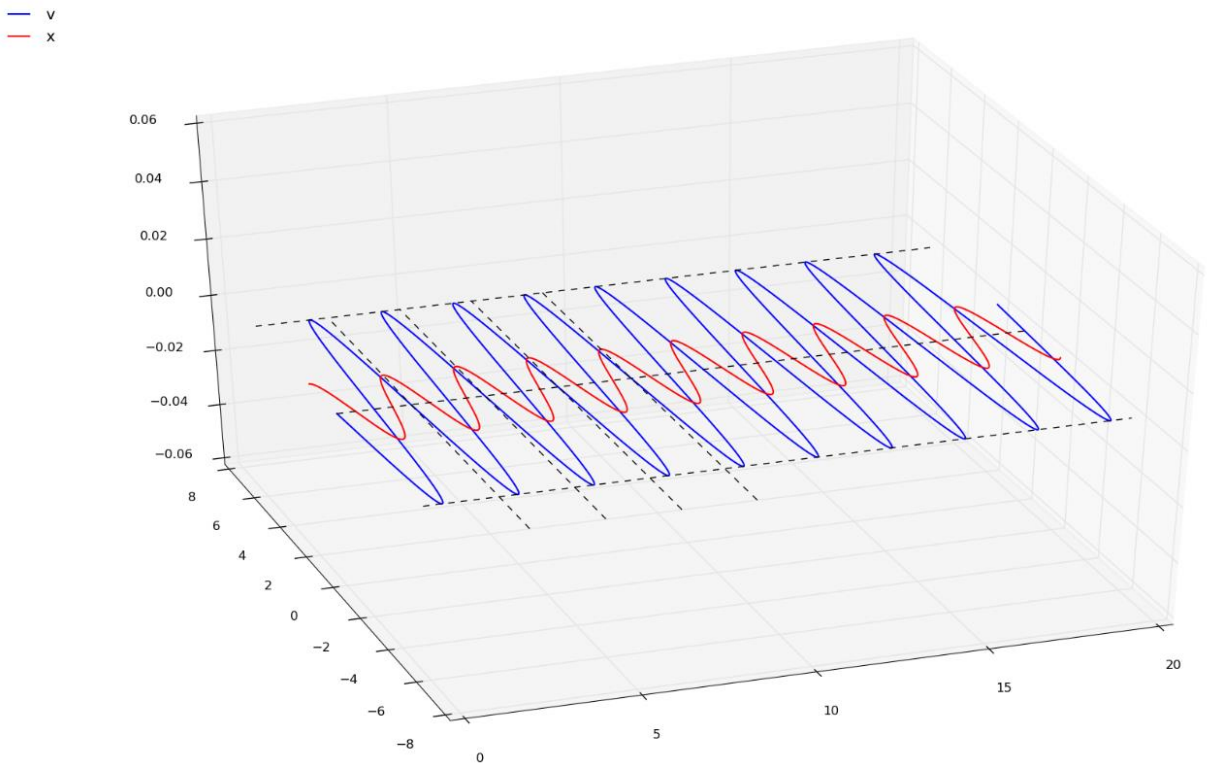
As it turns out,third correction changes almost nothing.Hence there is no need for higher order corrections here.

Finally,to be perfect, we choose third approximation and set $dt = 0.0001$, corresponding result is



It is excellent.

And we draw a 3D figure of it as



The code below is added to plot a 3D figure.

```
from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure()

ax = fig.add_subplot(111, projection='3d')
```

Conclusion:

While third and higher order corrections might be of no need in some cases, second order correction does make approximation better.For higher order differential equations,the value of time step in numerical computation matters much more.

As always,numerical computation is powerful.