Damped Driven Quartic Oscillator and Anharmonic Resonance

Consider a damped quartic oscillator, where the potential is $V = \frac{1}{2}\omega_0^2 x^2 + \frac{1}{4}\beta x^4$

So, the corresponding equation of motion will be: $\ddot{x} = -\omega_0^2 x - \gamma v - \beta x^3$

Now let's force this oscillator with driving force $F = f \sin \omega t$

So, now the equation of motion becomes: $\ddot{x} = -\omega_0^2 x - \gamma v - \beta x^3 + f_0 \sin \omega t$

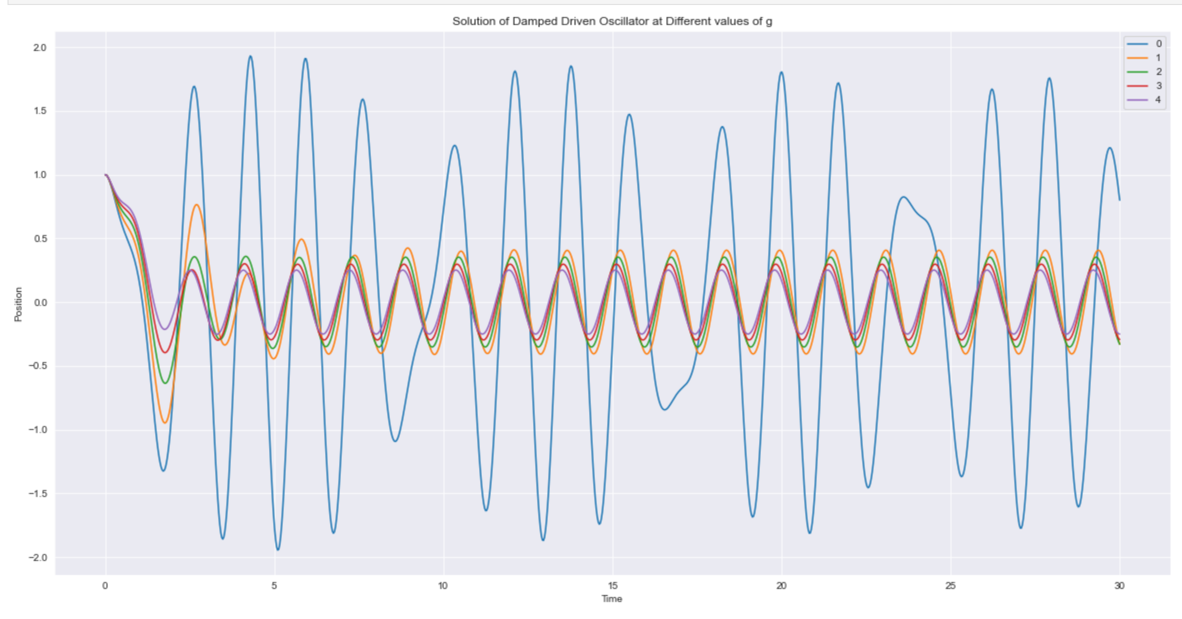
Now, let's write a program to get the solution of the above equation.

In [1]:

```
import matplotlib.pyplot as plt
import seaborn as sns
import math as mth
import numpy as np
sns.set_style('darkgrid')
```

In [7]:

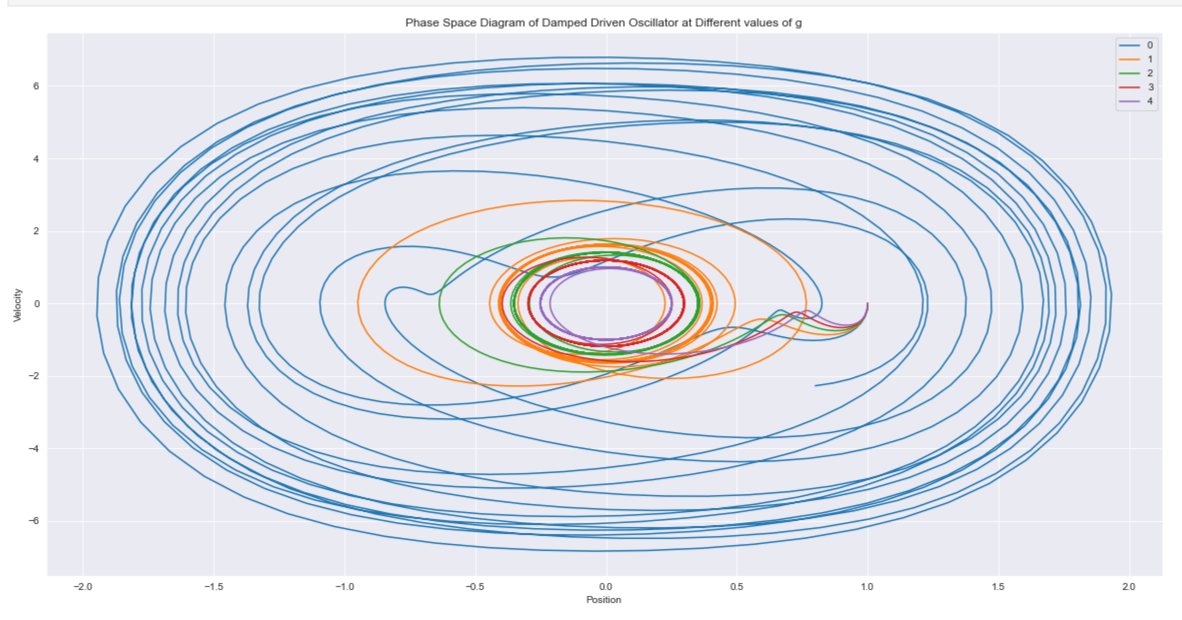
```
def f(t, x, v):
    return -w0**2*x - g*v - b* x**3 + f0 * mth.sin(w*t)

f0 = 5
w = 4
w0 = 2
b = 3
ti = 0
x0 = 1
v0 = 0
tf = 30
n = 1000
h = (tf - ti) / n
t = ti
x = x0
v = v0

for g in range(0,5):

    t = ti
    x = x0
    v = v0

    x_list = [x0]
    v_list = [v0]
    t_list = [ti]

    for i in range(n):
        #print(t,x,v)

        k1 = v
        j1 = f(t,x,v)

        k2 = v + (h*j1)/2.0
        j2 = f(t+ h/2.0, x+ (k1*h)/2.0, v + (j1*h)/ 2.0)

        k3 = v + (h*j2)/2.0
        j3 = f(t+ h/2.0, x+ (k2*h)/2.0, v + (j2*h)/ 2.0)

        k4 = v + (h*j3)
        j4 = f(t+ h, x+ (k3*h), v + (j3*h))

        x = x + h*(k1 + 2.0* k2 + 2.0* k3 + k4)/6.0
        v = v + h*(j1 + 2.0* j2 + 2.0* j3 + j4)/6.0
        t = t+h

        x_list.append(x)
        v_list.append(v)
        t_list.append(t)

    plt.plot(t_list, x_list, label = g)
    plt.legend()
    plt.rcParams["figure.figsize"] = (20, 10)
    plt.title("Solution of Damped Driven Oscillator at Different values of g")
    plt.xlabel("Time")
    plt.ylabel("Position")
```



Phase Space Diagram:

In [3]:

```
def f(t, x, v):
    return -w0**2*x - g*v - b* x**3 + f0 * mth.sin(w*t)

f0 = 5
w = 4
w0 = 2
b = 3
ti = 0
x0 = 1
v0 = 0
tf = 30
n = 1000
h = (tf - ti) / n
t = ti
x = x0
v = v0

for g in range(0,5):

    t = ti
    x = x0
    v = v0

    x_list = [x0]
    v_list = [v0]
    t_list = [ti]

    for i in range(n):
        #print(t,x,v)

        k1 = v
        j1 = f(t,x,v)

        k2 = v + (h*j1)/2.0
        j2 = f(t+ h/2.0, x+ (k1*h)/2.0, v + (j1*h)/ 2.0)

        k3 = v + (h*j2)/2.0
        j3 = f(t+ h/2.0, x+ (k2*h)/2.0, v + (j2*h)/ 2.0)

        k4 = v + (h*j3)
        j4 = f(t+ h, x+ (k3*h), v + (j3*h))

        x = x + h*(k1 + 2.0* k2 + 2.0* k3 + k4)/6.0
        v = v + h*(j1 + 2.0* j2 + 2.0* j3 + j4)/6.0
        t = t+h

        x_list.append(x)
        v_list.append(v)
        t_list.append(t)

    plt.plot(x_list, v_list, label = g)
    plt.legend()
    plt.rcParams["figure.figsize"] = (20, 10)
    plt.title("Phase Space Diagram of Damped Driven Oscillator at Different values of g")
    plt.xlabel("Position")
    plt.ylabel("Velocity")
```

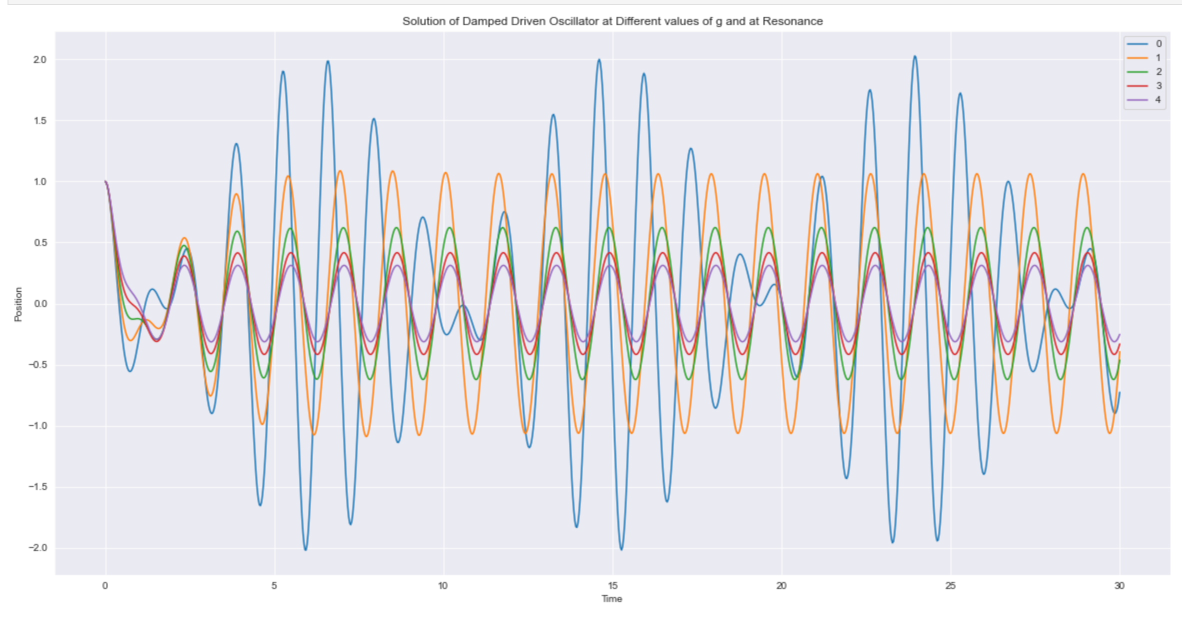


Resonance:

Now Let's see the Resonace Curves, when natural frequency of oscillation becomes equal to the driving frequency.

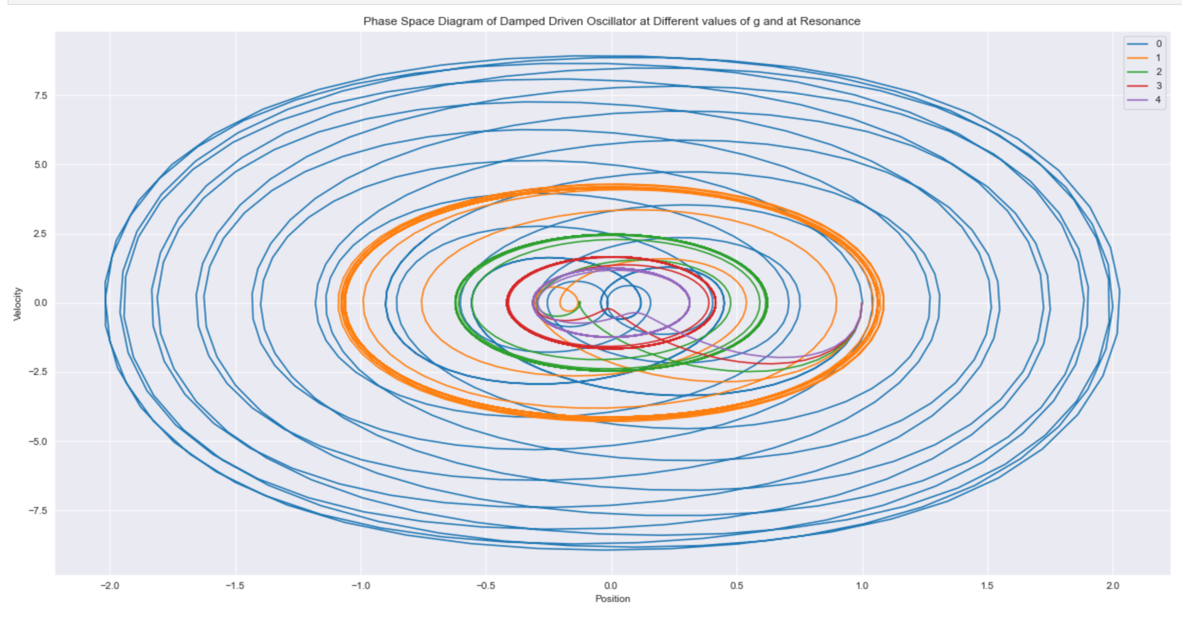
In [4]:

```
def f(t, x, v):
    return -w0**2*x - g*v - b* x**3 + f0 * mth.sin(w*t)

f0 = 5
w = 4
w0 = 4
b = 3
ti = 0
x0 = 1
v0 = 0
tf = 30
n = 1000
h = (tf - ti) / n
t = ti
x = x0
v = v0

for g in range(0,5):

    t = ti
    x = x0
    v = v0

    x_list = [x0]
    v_list = [v0]
    t_list = [ti]

    for i in range(n):
        #print(t,x,v)

        k1 = v
        j1 = f(t,x,v)

        k2 = v + (h*j1)/2.0
        j2 = f(t+ h/2.0, x+ (k1*h)/2.0, v + (j1*h)/ 2.0)

        k3 = v + (h*j2)/2.0
        j3 = f(t+ h/2.0, x+ (k2*h)/2.0, v + (j2*h)/ 2.0)

        k4 = v + (h*j3)
        j4 = f(t+ h, x+ (k3*h), v + (j3*h))

        x = x + h*(k1 + 2.0* k2 + 2.0* k3 + k4)/6.0
        v = v + h*(j1 + 2.0* j2 + 2.0* j3 + j4)/6.0
        t = t+h

        x_list.append(x)
        v_list.append(v)
        t_list.append(t)

    plt.plot(t_list, x_list, label = g)
    plt.legend()
    plt.rcParams["figure.figsize"] = (20, 10)
    plt.title("Solution of Damped Driven Oscillator at Different values of g and at Resonance")
    plt.xlabel("Time")
    plt.ylabel("Position")
```



In [5]:

```
def f(t, x, v):
    return -w0**2*x - g*v - b* x**3 + f0 * mth.sin(w*t)

f0 = 5
w = 4
w0 = 4
b = 3
ti = 0
x0 = 1
v0 = 0
tf = 30
n = 1000
h = (tf - ti) / n
t = ti
x = x0
v = v0

for g in range(0,5):

    t = ti
    x = x0
    v = v0

    x_list = [x0]
    v_list = [v0]
    t_list = [ti]

    for i in range(n):
        #print(t,x,v)

        k1 = v
        j1 = f(t,x,v)

        k2 = v + (h*j1)/2.0
        j2 = f(t+ h/2.0, x+ (k1*h)/2.0, v + (j1*h)/ 2.0)

        k3 = v + (h*j2)/2.0
        j3 = f(t+ h/2.0, x+ (k2*h)/2.0, v + (j2*h)/ 2.0)

        k4 = v + (h*j3)
        j4 = f(t+ h, x+ (k3*h), v + (j3*h))

        x = x + h*(k1 + 2.0* k2 + 2.0* k3 + k4)/6.0
        v = v + h*(j1 + 2.0* j2 + 2.0* j3 + j4)/6.0
        t = t+h

        x_list.append(x)
        v_list.append(v)
        t_list.append(t)

    plt.plot(x_list, v_list, label = g)
    plt.legend()
    plt.rcParams["figure.figsize"] = (20, 10)
    plt.title("Phase Space Diagram of Damped Driven Oscillator at Different values of g and at Resonance")
    plt.xlabel("Position")
    plt.ylabel("Velocity")
```



Amplitude Resonance Curves:

For this we need the steady state solution amplitude as a function of driving frequency.

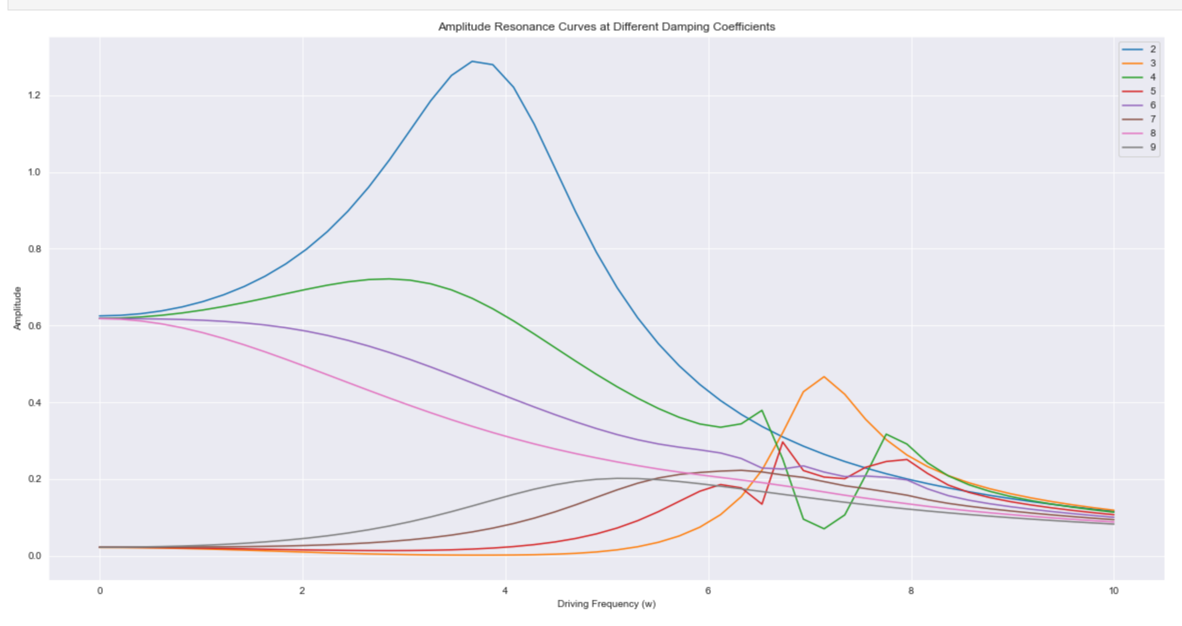
In the case of linear damped driven oscillator, we had it as $a = \frac{f_0}{\sqrt{(\omega^2 - \omega_0^2)^2 + (\gamma\omega)^2}}$

Where, f_0 is amplitude of driving force, ω is driving frequency, ω_0 is natiural frequency of oscillator and γ is damping constant. Now in this quartic oscillator there exists a correction to natural frequency ω_0, due to the extra x^4 term is potential. So, the corrected frequency is $\bar{\omega} = \omega_0 + \frac{3\beta a^2}{8\omega_0}$ or $\bar{\omega} = \omega_0 + \kappa a^2$. Where $\kappa = \frac{3\beta}{8\omega_0}$. So, to get the correct expression for amplitude, we just need to change ω_o to $\bar{\omega}$ in the above equation.

so, $a = \frac{f_0}{\sqrt{[\omega^2 - (\omega_0 + \kappa a^2)^2]^2 + (\gamma\omega)^2}}$ is the amplitude of steady state solution. Now let's plot this equation for different values of γ.

In [19]:

```
def ampl(f0, w, w0):
    return f0/np.sqrt((w**2 - (w0+k*A**2)**2)**2 + (g**2)*(w**2))

f0 = 10
k = 45
W0 = 4
A = 0
w = np.linspace(0, 10)

for g in range(2,10):
    A = ampl(f0, w, W0)
    plt.plot(w, A, label = g)
    plt.title("Amplitude Resonance Curves at Different Damping Coefficients")
    plt.legend()
    plt.xlabel("Driving Frequency (w)")
    plt.ylabel("Amplitude")
```



Yes, I think there is a region of instability from driving frequncy 6 onwards.