Exercise 3.1: Van der Pol oscillator

The van der Pol oscillator, which appears in electronic circuits and in laser physics, is described by the equation:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + \omega x^2 = 0$$

Write a program to solve this equation from t = 0 to t = 20 and hence make a phase space plot for the van der Pol oscillator with $\omega = 1$, $\mu = 1$, and initial conditions x = 1 and dx/dt = 0. Try it also for $\mu = 2$ and µ = 4 (still with $\omega = 1$). Make sure you use a small enough value of the time interval h to get a smooth, accurate phase space plot.

Given:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + \omega^2 x = 0$$

with:

$$x = 1, \frac{dx}{dt} = 0, \omega = 1$$

We have:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = 0$$

Transfer above equation to a system of first order ODE:

$$\dot{x} = v$$

$$\dot{v} = -x - \mu(x^2 - 1)v$$

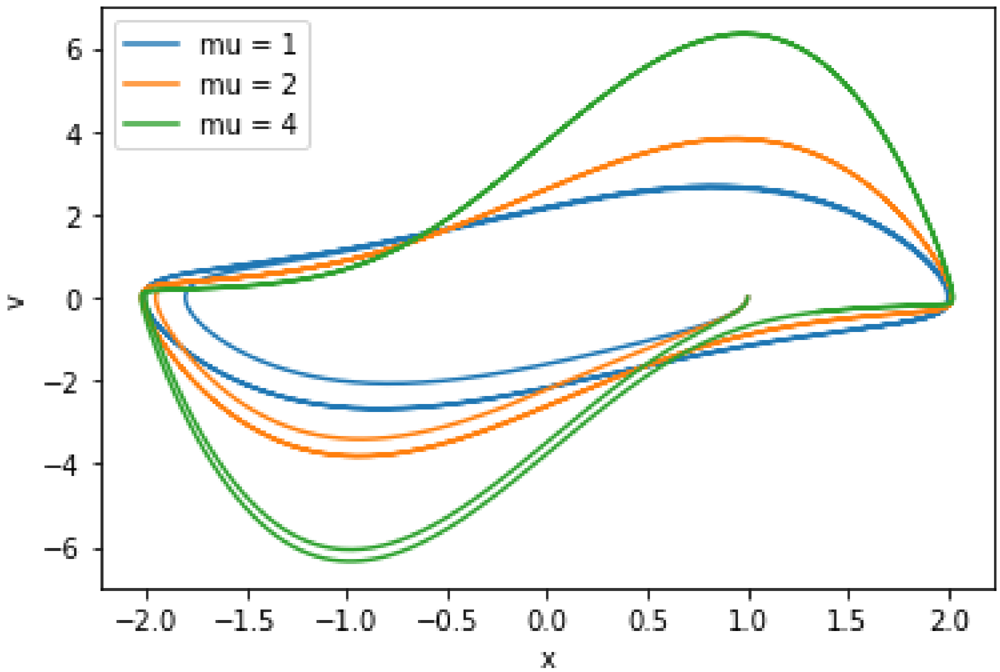
Here I use RK4 to solve this problem.

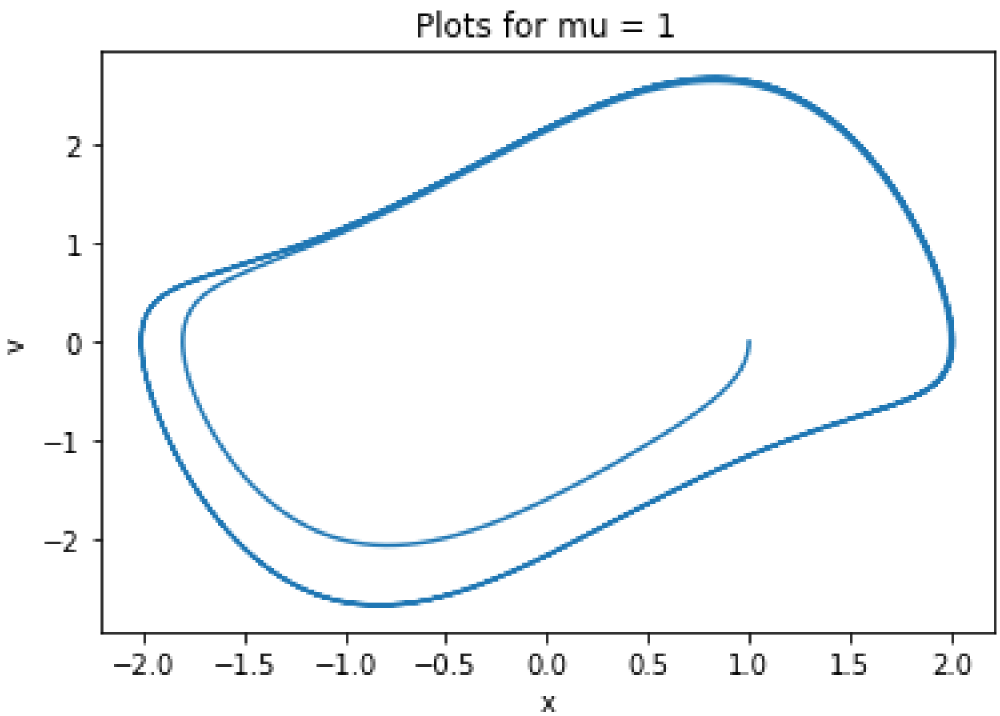
Python code:

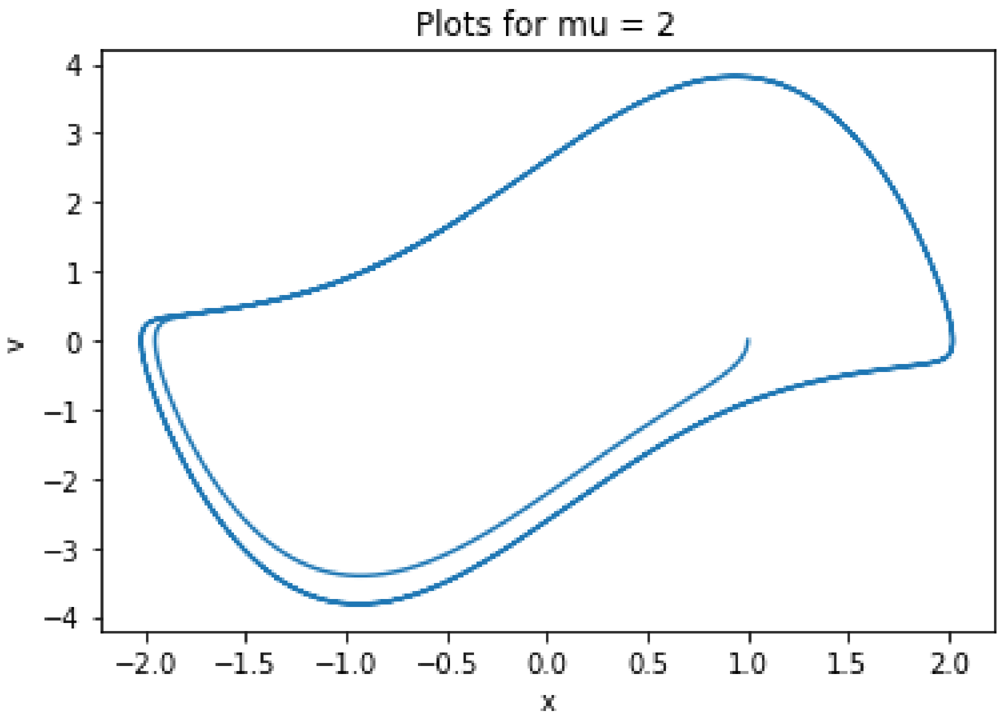
```
# -*- coding: utf-8 -*-
"""
@author: JH_Song
"""

from numpy import array, arange
from matplotlib import pyplot as plt

# Constants
t_0 = 0
t_f = 20
N = 10000  # number of steps
h = (t_f - t_0) / N

#Initial conditions
x_0 = 1
v_0 = 0

#function
def g(r, t, mu):
    x = r[0]
    v = r[1]
    return array([v, -x - mu*(x**2 - 1)*v], float)
```

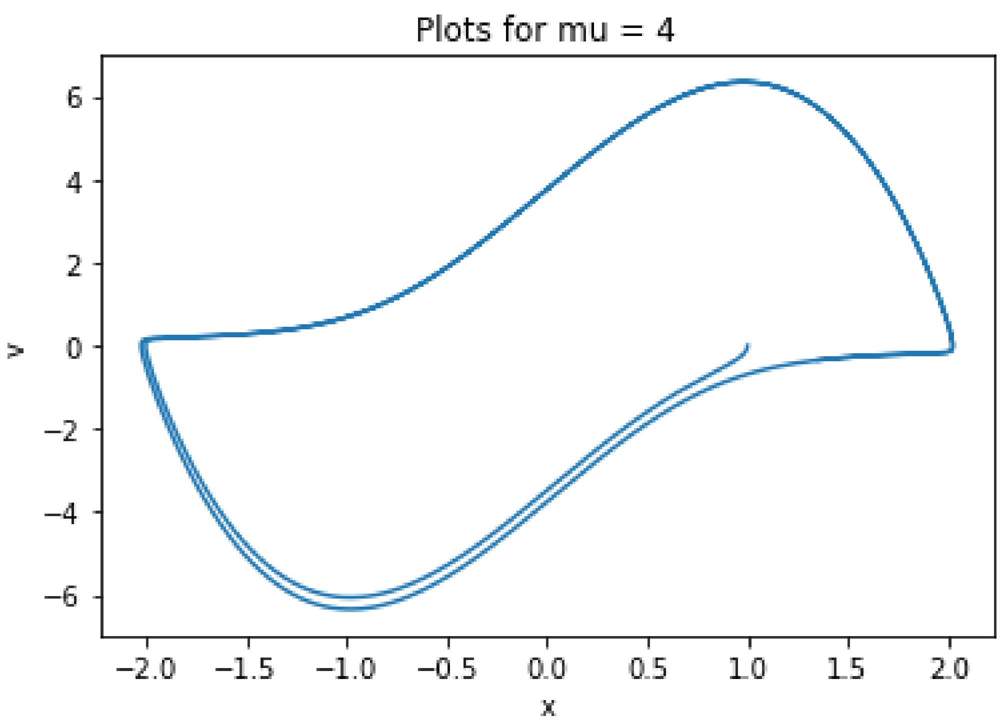
```
#RK-4 solver
tpoints = arange(t_0, t_f, h)
def VanDerPol_RK4(mu):
    xpoints = []
    vpoints = []
    r = array([x_0, v_0], float)
    for t in tpoints:
        xpoints.append(r[0])
        vpoints.append(r[1])
        k1 = h * g(r, t, mu)
        k2 = h * g(r + 0.5 * k1, t + 0.5 * h, mu)
        k3 = h * g(r + 0.5 * k2, t + 0.5 * h, mu)
        k4 = h * g(r + k3, t + h, mu)
        r += (k1 + 2 * k2 + 2 * k3 + k4) / 6
    return array(xpoints, float), array(vpoints, float)

# 3 in 1 plot
x1, v1 = VanDerPol_RK4(1) #plot for mu = 1
x2, v2 = VanDerPol_RK4(2) #plot for mu = 2
x3, v3 = VanDerPol_RK4(4) #plot for mu = 4
plt.plot(x1, v1, label='mu = 1')
plt.plot(x2, v2, label='mu = 2')
plt.plot(x3, v3, label='mu = 4')
plt.xlabel('x')
plt.ylabel('v')
plt.legend()
plt.show()

# 3 invividual plot
plt.plot(x1, v1)
plt.xlabel('x')
plt.ylabel('v')
plt.title("Plots for mu = 1")
plt.show()
plt.plot(x2, v2)
plt.xlabel('x')
plt.ylabel('v')
plt.title("Plots for mu = 2")
plt.show()
plt.plot(x3, v3)
plt.xlabel('x')
plt.ylabel('v')
plt.title("Plots for mu = 4")
plt.show()
```

Sample output:






Plots for mu = 2
y
x


Plots for mu = 4
y
x