# 14 Sept 23 - Conducting a complete analysis of an ODE

We’ve built up all the tools that we use to investigate ODEs. Now we are going to practice with an oscillator that exhibits really interesting dynamics: the [Van der Pol oscillator](https://en.wikipedia.org/wiki/Van_der_Pol_oscillator). This equation originates from nonlinear circuits in early radios, but has now also been used in neuroscience and geology. It is given by the differential equation:

or, written as two first order equations:

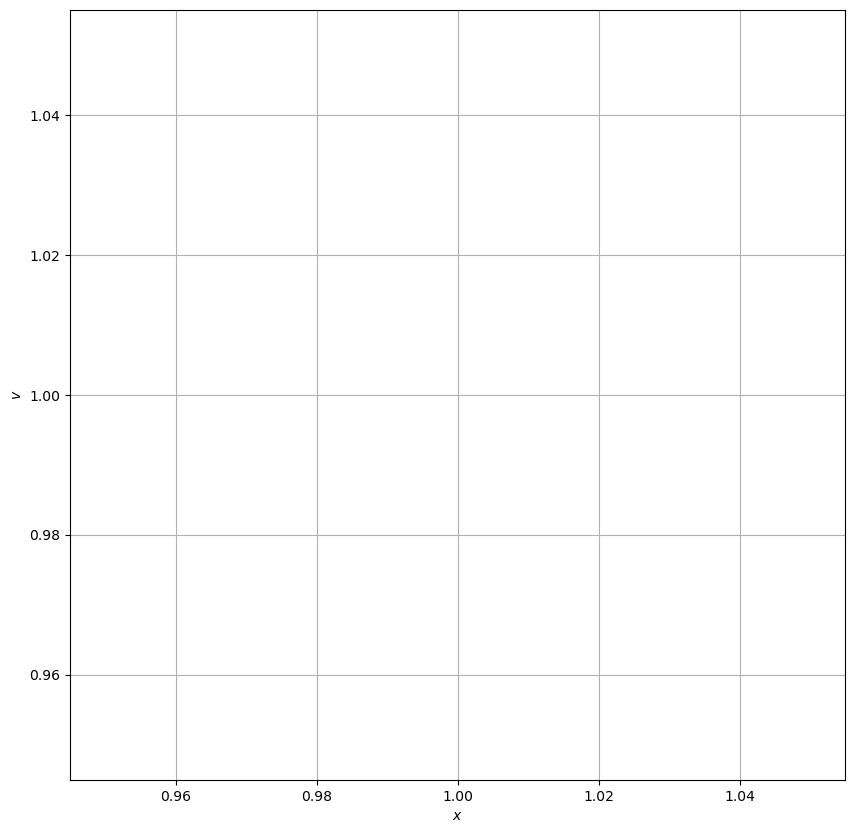
With . Note that this equation is simply the harmonic oscillator when . The strange represents damping, but this damping behaves strangely, because when it is negative damping, that is it boosts oscillations smaller than , while still slowing down oscillations larger than .

Now we play the usual game of trying to figure out how this system behaves:

**✅ Do this**

1. Identify the fixed point of this system. Follow the linearization procedure to characterize it.
2. Edit the code below to produce a phase plot for the Van der Pol oscillator. This code also numerically integrates a trajectory and plots it. Add a second trajectory and plot that as well.
3. What happens to phase space when you change the value of ? What if you make it negative?
4. What behavior do you notice here that’s different than you’ve seen before? What is attracting the trajectories?

import numpy as np  
import matplotlib.pyplot as plt  
from scipy.integrate import solve\_ivp  
  
def VP\_eqn(x, v, mu = 1.):  
 xdot, vdot = [0,0] ## CHANGE  
 return xdot, vdot  
  
def VP\_phase(X, VX, mu):  
 xdot, vdot = np.zeros(X.shape), np.zeros(VX.shape)  
 Xlim, Ylim = X.shape  
 for i in range(Xlim):  
 for j in range(Ylim):  
 xloc = X[i, j]  
 yloc = VX[i, j]  
 xdot[i,j], vdot[i,j] = VP\_eqn(xloc, yloc,mu)  
 return xdot, vdot  
  
def VP\_eqn\_for\_solve\_ivp(t,curr\_vals, mu=1): # need to rephrase this to work with what solve\_ivp expects  
 x, v = curr\_vals   
 xdot, vdot = VP\_eqn(x,v,mu)  
 return xdot,vdot  
  
# Numerical Integration  
tmax = 20  
dt = 0.05  
tspan = (0,tmax)  
t = np.arange(0,tmax,dt)  
mu = 1.  
initial\_condition = [1, 1]   
solved = solve\_ivp(VP\_eqn\_for\_solve\_ivp,tspan,initial\_condition,t\_eval = t, args = (mu,),method="RK45")  
  
  
# Plotting stuff  
N = 40  
x = np.linspace(-3., 3., N)  
v = np.linspace(-3., 3., N)  
X, V = np.meshgrid(x, v)  
xdot, vdot = VP\_phase(X, V,mu)  
ax = plt.figure(figsize=(10,10))  
Q = plt.streamplot(X, V, xdot, vdot, color='k',broken\_streamlines = False)  
plt.plot(solved.y[0],solved.y[1],lw = 3,c = 'red') # plot trajectory from solve\_ivp  
plt.grid()  
plt.xlabel('$x$')  
plt.ylabel('$v$')  
plt.show()



png

**✅ Do this**

Based on the phase space diagram, what do you expect actual trajectories to look like in vs space? Use the numerically integrated trajectories to plot that.

## Limit Cycles

The new behavior we’ve seen from this equation is what’s called a **limit cycle**, where the system is attracted/reppeled from a closed curve instead of a fixed point(s). There’s a lot of really great math here that’s a bit beyond what we can cover in class, but it would be a great thing to look into for a project!

**✅ Do this**

Spend the rest of class investigating the Van der Pol oscillator. Here are a few investigations you could do:

* When changes from negative to positive, this system undergoes what is known as a **Hopf Bifurcation** Look up what bifurcations are to understand what this means and show that it is true using numerical integration.
* Add an driving force term to the differential equation and numerically integrate. What do these trajectories look like in vs and in phase space?
* Examine the energetics of this system. Is energy conserved or does it have some interesting behavior? Why?