0.1 /code

/code

```
#coding=utf-8

import matplotlib
matplotlib.use(’ps’)
import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import solve_ivp


def plot_phase_trajectories(f, inits, xbound, ybound, t=(0, 10), steps=100,
                            axis=None,
                            sivp_kwargs={}, plt_kwargs={’c’: ’b’, ’lw’: .7}):
    """Plots phase trajectory of given ODE with scipy.integrate.solve_ivp
    Returns list of matplotlib-artist objects"""
    if axis is None:
        axis = plt.gca()

    def f_neg(t, x):
        # for solution backwards in time
        return -f(t, x)

    artists = []
    tt = np.linspace(*t, steps)
    for ff in (f, f_neg):
        for i in inits:
            # solve_ivp(..., dense_output=True).sol holds a OdeSolution object
            # which interpolates the solution and allows its evaluation at
            # arbitrary points
            # Returns array with shape(n,) corresponding to the RHS of the
            # given ODE
            sol = solve_ivp(ff, t, i, dense_output=1, **sivp_kwargs).sol
            sol_eval = sol(tt)
            sol_x_ma = np.ma.masked_outside(sol_eval[0], *xbound)
            sol_y_ma = np.ma.masked_outside(sol_eval[1], *ybound)
            artists.append(axis.plot(sol_x_ma, sol_y_ma, **plt_kwargs))

    plt.xlim(xbound)
    plt.ylim(ybound)
    return artists


ppt = plot_phase_trajectories

```

```
inits = np.array([(i, j) for i in np.arange(-10, 11, 2) for j in np.arange(-1.5, 2, 1)])

def funcs(s, n):
    def func(t, x):
        """f(t, x) = s*x(t)**n * (1 - x(t)**2)"""
        return np.array([1, s*x[1]**n*(1-x[1]**2)])
    return func

fs = [(funcs(s, n), (s, n)) for s in (1, -1) for n in (1, 2)]



def make_plot(name='plot', filetype='ps'):
    f, axarr = plt.subplots(4, sharex=True, figsize=(8, 12))
    for i in range(4):
        plt.subplot(4, 1, i+1)
        ppt(fs[i][0], inits, (-5, 5), (-1.5, 1.5))
        plt.title('s = {0}, n = {1}'.format(*fs[i][1]))
        plt.ylabel('x')
    plt.setp([a.get_xticklabels() for a in f.axes[:-1]], visible=False)
    plt.xlabel('t')
    plt.tight_layout()
    plt.savefig(name+'.'+filetype, dpi=100)
```