PYTHON EXAMPLES

WILLIAM BLACK

1. Streamplots

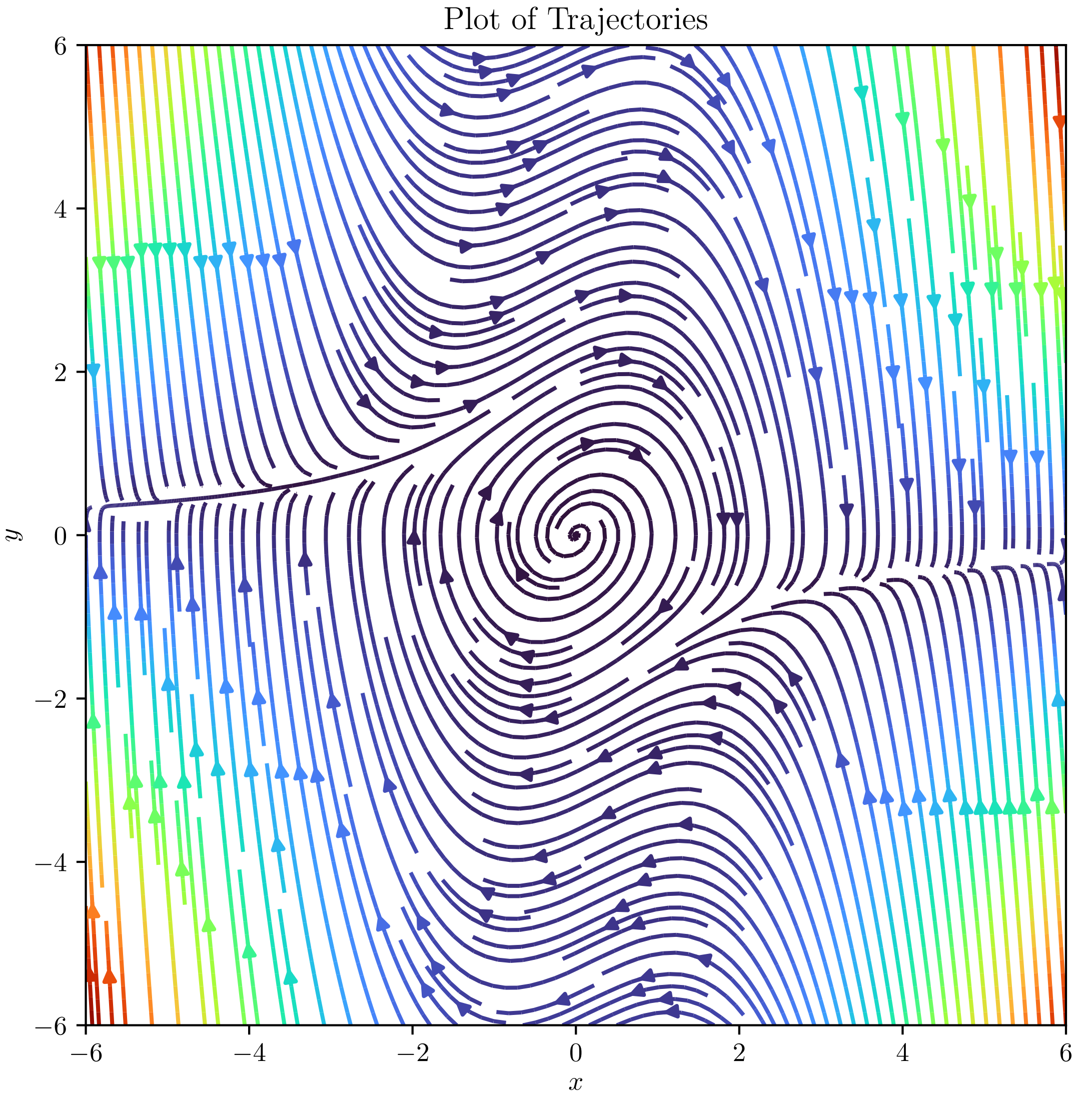
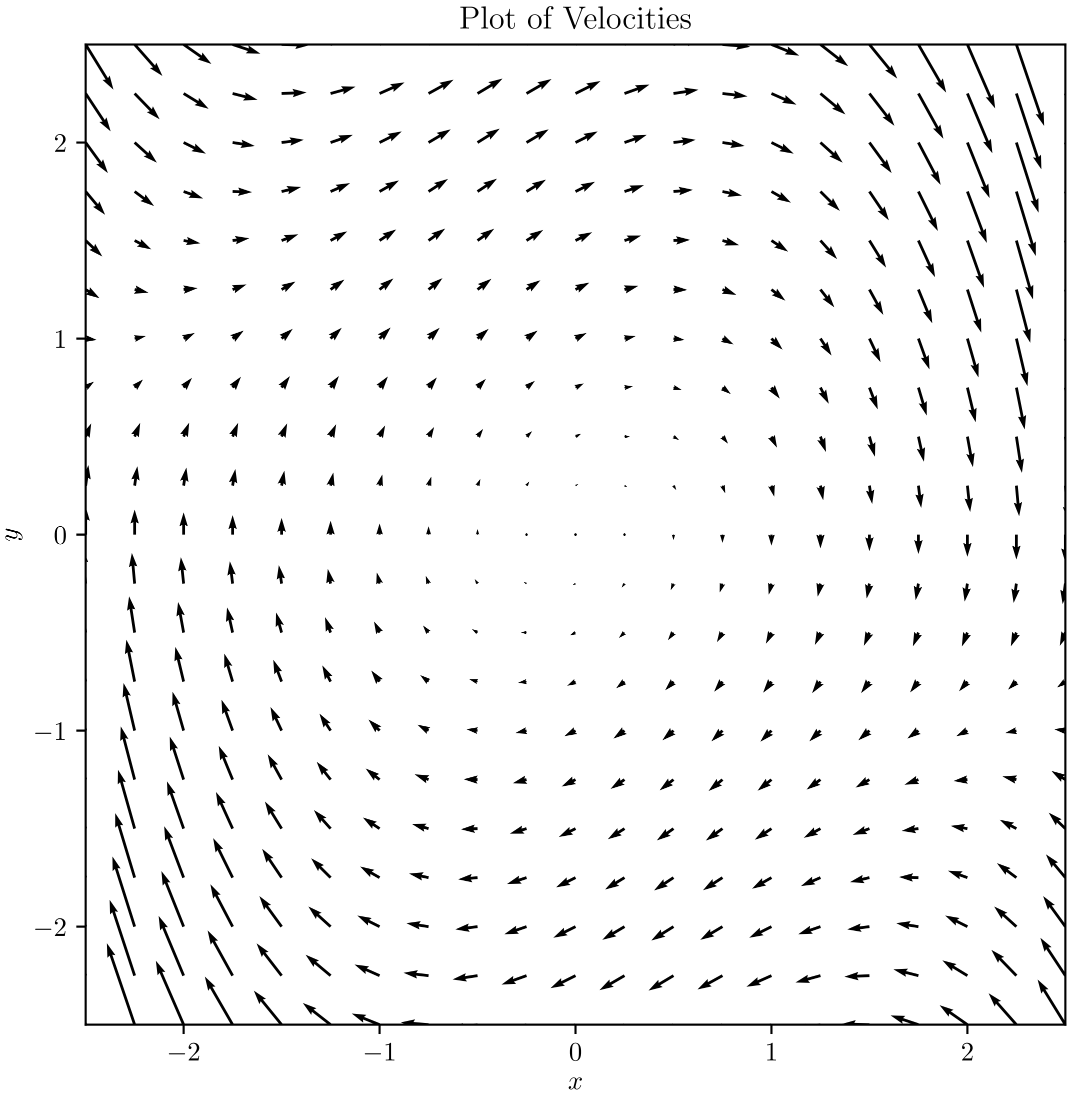


Figure 1. This is a plot of the Van der Pol system with $\epsilon = 0.5$. Note that the pgf format creates vector images that can be zoomed.

```
"""
streamplot.py
By William Black

This script outputs a streamplot for the Van der Pol system to pgf to be used in LaTeX.
"""
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import norm

plt.rcParams.update({ ## This will make the plots render in native LaTeX in your PDF.
    "text.usetex": True,
    "font.family": "serif",
    "pgf.texsystem" : "pdflatex",
    "axes.unicode_minus" : False,
    "text.latex.preamble" : r"\usepackage{amsmath,amsfonts,amssymb,mathtools}"
})

def vanderpol(x, y, eps=1.): ## Default value of 1. for eps
    return np.stack([y,
                     eps * (1 - x**2) * y - x])


if __name__ == "__main__":
    xlim = (-6., 6.)
    ylim = (-6., 6.)
    N = 20 ## Number of gridpoints per dimension

    X = np.linspace(*xlim, num=1+N) ## Using the unpacking operator *
    Y = np.linspace(*ylim, num=1+N)

    Xg, Yg = np.meshgrid(X, Y) ## g for grid

    Xv, Yv = vanderpol(Xg, Yg, eps=0.5)

    color = norm(np.stack([Xv, Yv]), ord=2, axis=0)

    plt.figure(figsize=(6., 6.))
    plt.gca().set_aspect("equal", adjustable="box")

    plt.xlim(*xlim)
    plt.ylim(*ylim)

    plt.streamplot(
        X,
        Y,
        Xv,
        Yv,
        density=2.5,
        arrowsize=1.,
        color=color,
        cmap=plt.get_cmap("turbo")
    )

    plt.xlabel("$x$")
    plt.ylabel("$y$")
    plt.title("Plot␣of␣Trajectories")
    plt.tight_layout()
    plt.savefig("streamplot.pgf") ## You can change pgf to png or other formats
```

2. Velocity Plots



```python
"""
velocityplot.py
By William Black

This script outputs a velocityplot for the Van der Pol system to pgf to be used in LaTeX.
"""
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import norm

plt.rcParams.update({ ## This will make the plots render in native LaTeX in your PDF.
    "text.usetex": True,
    "font.family": "serif",
    "pgf.texsystem" : "pdflatex",
    "axes.unicode_minus" : False,
    "text.latex.preamble" : r"\usepackage{amsmath,amsfonts,amssymb,mathtools}"
})

def vanderpol(x, y, eps=1.): ## Default value of 1. for eps
    return np.stack([y,
                     eps * (1 - x**2) * y - x])


if __name__ == "__main__":
    xlim = (-2.5, 2.5)
    ylim = (-2.5, 2.5)
    N = 20 ## Number of gridpoints per dimension

    X = np.linspace(*xlim, num=1+N) ## Using the unpacking operator *
    Y = np.linspace(*ylim, num=1+N)

    Xg, Yg = np.meshgrid(X, Y) ## g for grid

    Xv, Yv = vanderpol(Xg, Yg, eps=0.5) ## v for velocity vector

    plt.figure(figsize=(6., 6.))
    plt.gca().set_aspect("equal", adjustable="box")

    plt.xlim(*xlim)
    plt.ylim(*ylim)

    plt.quiver(
        X,
        Y,
        Xv,
        Yv,
        units="xy"
    )

    plt.xlabel("$x$")
    plt.ylabel("$y$")
    plt.title("Plot␣of␣Velocities")
    plt.tight_layout()
    plt.savefig("velocityplot.pgf") ## You can change pgf to png or other formats
```