2023-05-15_ChartTypes_013_VectorFields-StreamPlots

May 25, 2023

[1]:
```python
import numpy as np
import scipy
import imageio

import matplotlib
import matplotlib.pyplot as plt
import matplotlib.cm as cm

matplotlib.rc('image', interpolation='nearest')
matplotlib.rc('figure',facecolor='white')
matplotlib.rc('image',cmap='viridis')
colors=plt.rcParams['axes.prop_cycle'].by_key()['color']
%matplotlib inline
```

1 Vector fields

1.1 Stream plots

- given vector field $v : \mathbb{R}^2 \to \mathbb{R}^2$, quiver plot shows small arrow $v(x)$ at positions x
- in some cases v is velocity field, e.g. wind in the air; a leaf would be taken along by the wind, its path is described by the equation

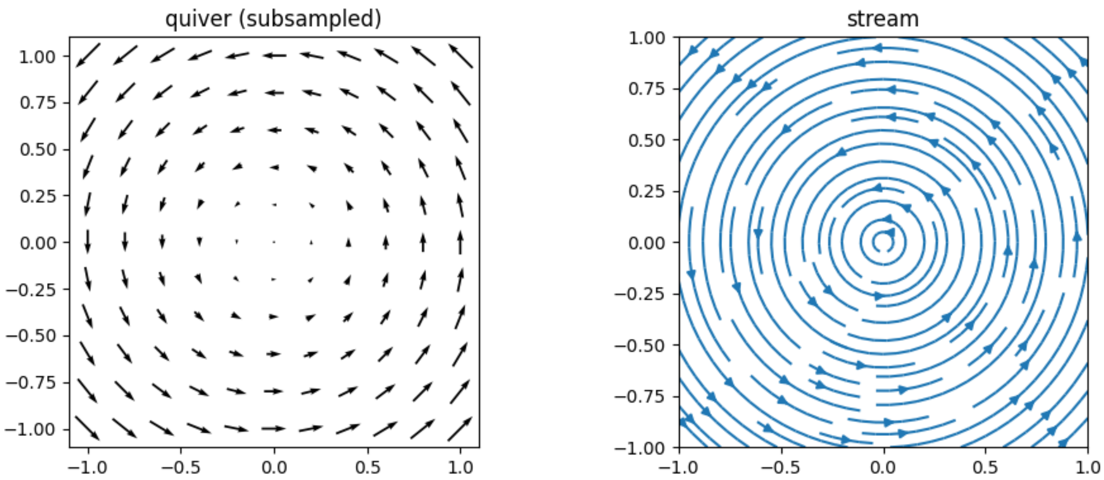
$$\partial_t y(t) = v(y(t)) \qquad \text{with some starting point } y(0)$$

- so instead of showing small arrows v at many positions, an alternative is to show paths of leafs for various starting points
- this is the idea of the stream plot

1.1.1 Simple circular example

[3]:
```python
nPts1d=21
x = np.linspace(-1,1,num=nPts1d)
y = x
nPts=nPts1d**2
X, Y = np.meshgrid(x, y)
u = -Y
```

```
v = X

fig=plt.figure(figsize=(10,4))
ax=fig.add_subplot(1,2,1,aspect=1.)
ax.quiver(X[::2,::2],Y[::2,::2],u[::2,::2],v[::2,::2],pivot="mid")
plt.title("quiver (subsampled)")

ax=fig.add_subplot(1,2,2,aspect=1.)
ax.streamplot(X,Y,u,v)
plt.title("stream")

plt.tight_layout()
plt.show()
```

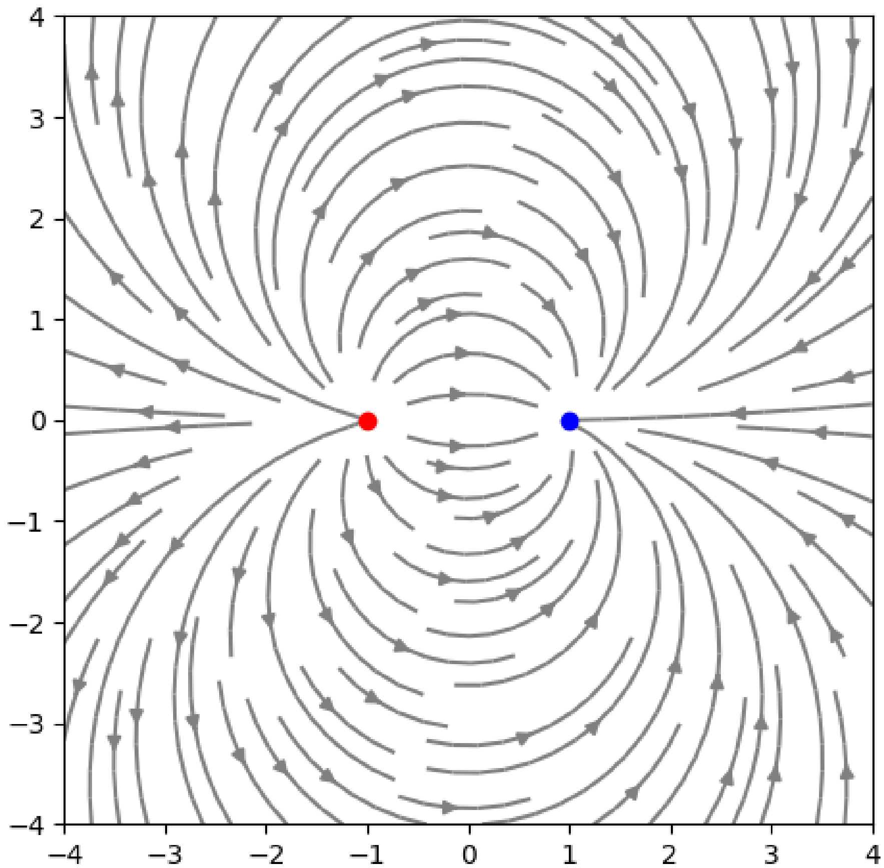


[4]:
```
# nice illustration of curl of a div free vector field:
# lines do not "collide", converge or diverge
```

1.1.2 Electric field of point charges

[5]:
```
# this example is inspired by:
# https://problemsolvingwithpython.com/06-Plotting-with-Matplotlib/06.
↪15-Quiver-and-Stream-Plots/
# Coulomb force field of two point charges wit opposite sign

nPts1d=200
x = np.linspace(-4,4,num=nPts1d)
y = x
nPts=nPts1d**2
```

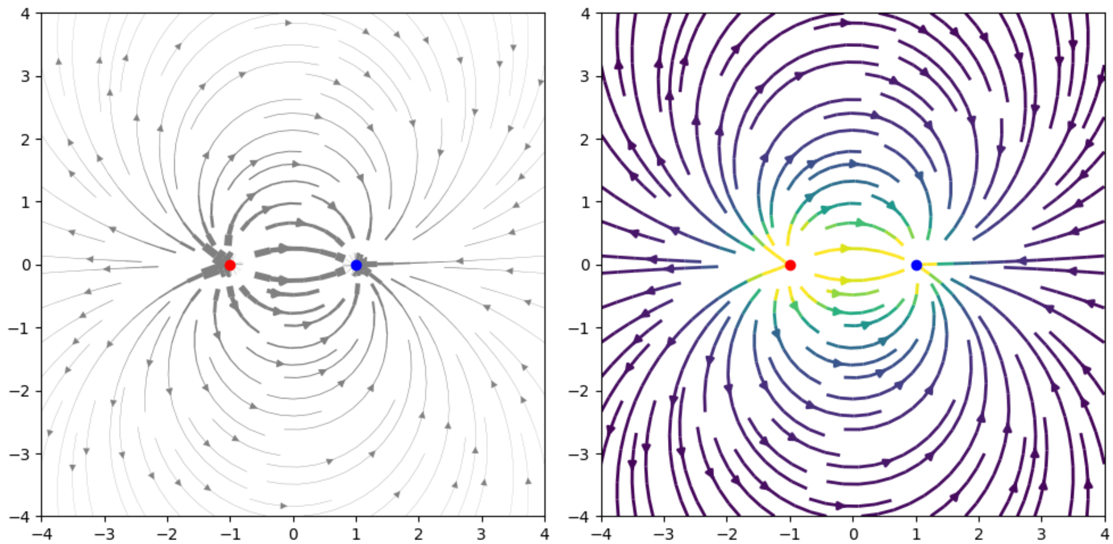
```
X,Y = np.meshgrid(x,y)
Ex = (X + 1)/((X+1)**2 + Y**2) - (X - 1)/((X-1)**2 + Y**2)
Ey = Y/((X+1)**2 + Y**2) - Y/((X-1)**2 + Y**2)

fig=plt.figure()
ax=fig.add_subplot(aspect=1.)

ax.streamplot(X,Y,Ex,Ey,color="#808080")

ax.scatter([-1],[0],c="r",zorder=2)
ax.scatter([1],[0],c="b",zorder=2)

plt.tight_layout()

plt.show()
```



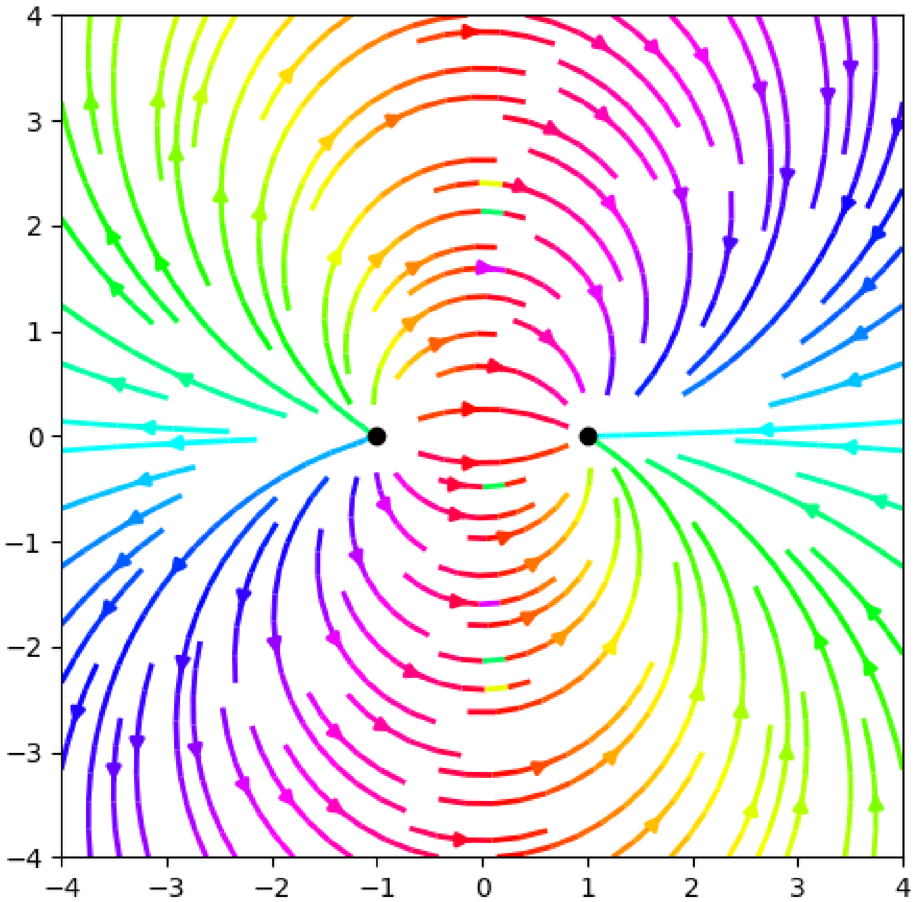
[6]:
```
# we see the divergence structure of the field relatively nicely
# there are clear sources and sinks of lines
```

1.1.3 Speed / field strength as color or thickness

[7]:
```
# this example is inspired by:
# https://problemsolvingwithpython.com/06-Plotting-with-Matplotlib/06.
↪15-Quiver-and-Stream-Plots/

nPts1d=100
x = np.linspace(-4,4,num=nPts1d)
y = x
nPts=nPts1d**2

X,Y = np.meshgrid(x,y)
Ex = (X + 1)/((X+1)**2 + Y**2) - (X - 1)/((X-1)**2 + Y**2)
Ey = Y/((X+1)**2 + Y**2) - Y/((X-1)**2 + Y**2)

EMag=(Ex**2+Ey**2)**0.5

fig=plt.figure(figsize=(10,5))

# line width
ax=fig.add_subplot(1,2,1,aspect=1.)
# signal: normalize, truncate at some maximal strength
vmax=10.
signal=np.clip(EMag/vmax,0.,1.)
ax.streamplot(X,Y,Ex,Ey,linewidth=15*signal,color="#808080")
ax.scatter([-1],[0],c="r",zorder=2)
ax.scatter([1],[0],c="b",zorder=2)


# line width
ax=fig.add_subplot(1,2,2,aspect=1.)
# signal: normalize, truncate at some maximal strength

vmax=2.
#signal=1-np.clip(EMag/vmax,0.,1.)
#ax.streamplot(X,Y,Ex,Ey,linewidth=2,color=1-signal,cmap=cm.gray)
signal=np.clip(EMag/vmax,0.,1.)
ax.streamplot(X,Y,Ex,Ey,linewidth=2,color=signal,cmap=cm.viridis)

ax.scatter([-1],[0],c="r",zorder=2)
ax.scatter([1],[0],c="b",zorder=2)

plt.tight_layout()
plt.show()
```



[8]:
```
# yet another one: color for orientation
fig=plt.figure(figsize=(5,5))

ax=fig.add_subplot(aspect=1.)

signal=np.mod(np.arctan2(Ey,Ex)/2/np.pi,1.)
ax.streamplot(X,Y,Ex,Ey,linewidth=2,color=signal,cmap=cm.hsv)

ax.scatter([-1,1],[0,0],c="k",zorder=2)

plt.tight_layout()
plt.show()
```



[ ]:
```
# can see some artifacts from linear interpolation around cut-off
# for "production" level figures probably need to implement this more carefully


# to be honest: most of these additional gimmicks seem to be "ducks" in these
  ↪examples
```