Rotacionais e gradientes

3 de Abril de 2020

Podemos entender o que ocorre ao calcularmos o rotacional de um campo da seguinte maneira: suponha um campo $\mathbf{A} = xy(\hat{x} + \hat{y} + \hat{z})$. Primeiramente, definimos o campo:

```
import numpy as np
from sympy import *
from sympy.physics.vector import ReferenceFrame, curl, divergence, gradient

S = ReferenceFrame('S')
A = (S[0]*S[1])*(S.x+S.y+S.z)
vars = [S[0],S[1],S[2]]
A
```

$S_x S_y \hat{\mathbf{s}}_{\mathbf{x}} + S_x S_y \hat{\mathbf{s}}_{\mathbf{y}} + S_x S_y \hat{\mathbf{s}}_{\mathbf{z}}$

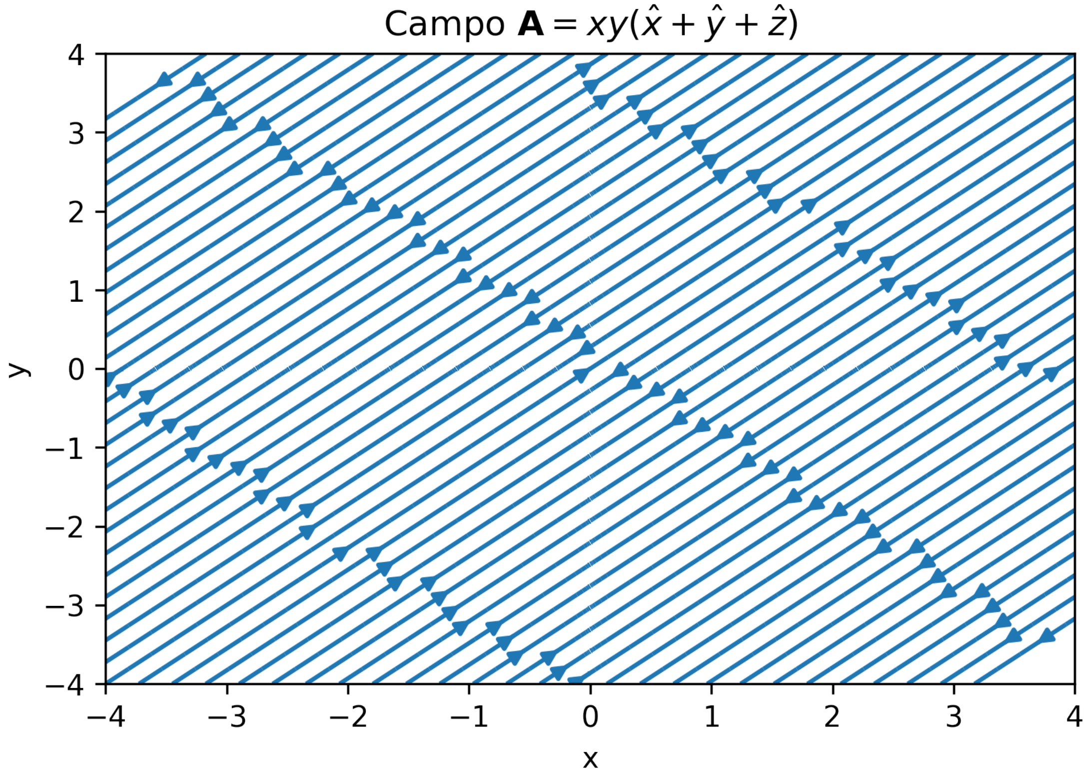
Veja que podemos calcular o rotacional de $\mathbf{A}$ referente ao referencial S com `curl(A,S)`:

```
curl(A,S)
```

Desejamos plotar o campo $\mathbf{A}$ e seu rotacional num plano $z = 0$.

```
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
figure(dpi=300)

X,Y = np.mgrid[-4:4:100j,-4:4:100j]
U = lambdify(vars,A.dot(S.x),modules='numpy')
V = lambdify(vars,A.dot(S.y),modules='numpy')
plt.streamplot(Y,X, U(X,Y,0), V(X,Y,0))
plt.title(r'Campo $\mathbf{A} = xy(\hat{x}+\hat{y}+\hat{z})$')
plt.xlabel(r'x')
plt.ylabel(r'y')
plt.show()
```



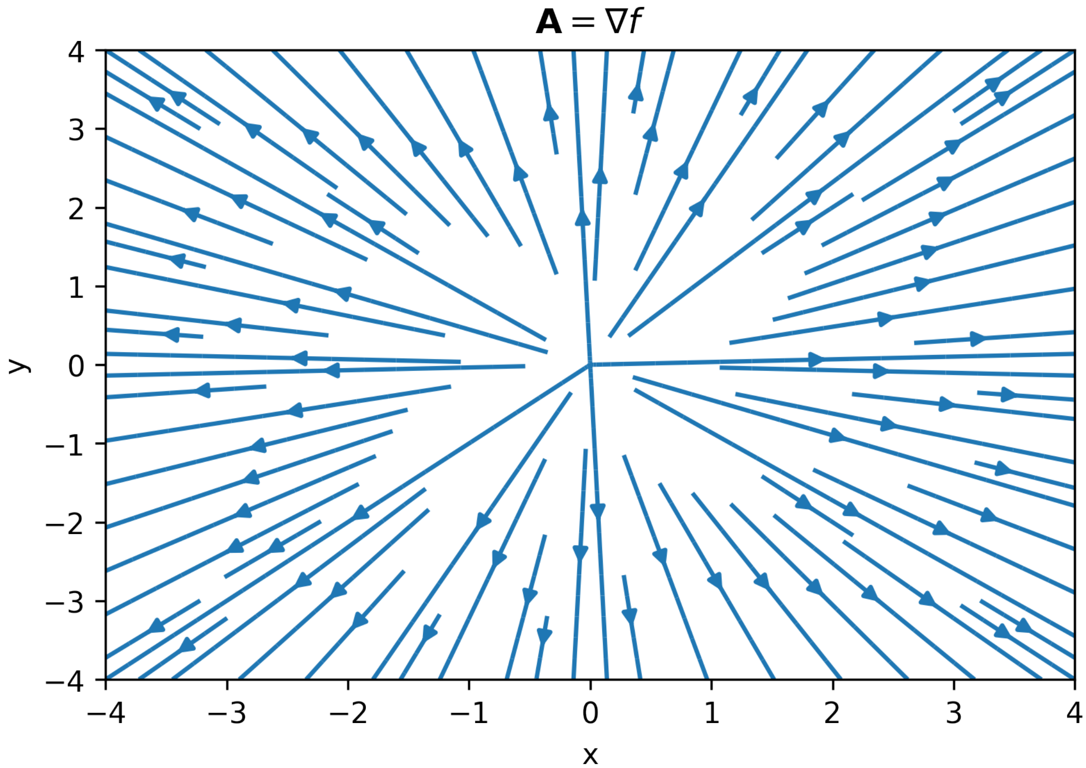

```
U = lambdify(vars,curl(A,S).dot(S.x),modules='numpy')
V = lambdify(vars,curl(A,S).dot(S.y),modules='numpy')
figure(dpi=300)
plt.streamplot(Y,X, U(X,Y,0), V(X,Y,0))
plt.title(r'$\nabla\times\mathbf{A}$')
plt.xlabel(r'x')
plt.ylabel(r'y')
plt.show()
```

Se tivéssemos definido o campo $\mathbf{A}$ como um gradiente de um campo escalar $f = xy$,

```
f = S[0]*S[1]
A = gradient(f,S)
A
```

$S_y\hat{\mathbf{s}}_{\mathbf{x}} + S_x\hat{\mathbf{s}}_{\mathbf{y}}$

```
U = lambdify(vars,A.dot(S.x),modules='numpy')
V = lambdify(vars,A.dot(S.y),modules='numpy')
figure(dpi=300)
plt.streamplot(Y,X,U(X,Y,0), V(X,Y,0))
plt.title(r'$\mathbf{A}=\nabla f$')
plt.xlabel(r'x')
plt.ylabel(r'y')
plt.show()
```



De tal modo que $\nabla \times A = 0$:

```
curl(A,S)
```

0

1 Integration

```
import matplotlib.pyplot as plt
import numpy as np
from sympy import *
from sympy.physics.vector import ReferenceFrame
from sympy.vector import Del

x,y,z = symbols('x y z')
nabla = Del()
S = ReferenceFrame('S')
r_vec = sum([S[indice]*versor for indice,versor in enumerate(S)])
r = r_vec.magnitude()
E = r_vec/r**3
dx = (S.x+S.y+S.z)

Integral(Integral(Integral(-E.dot(dx),(S[0],oo,x)),(S[1],oo,y)),(S[2],oo,z)).doit()
```

$$\int\limits_{\infty}^{z} \left(- \int\limits_{\infty}^{y} \frac{S_y x}{\sqrt{x^2 + \operatorname{polar\_lift}\left(S_y^2 + S_z^2\right)} \operatorname{polar\_lift}\left(S_y^2 + S_z^2\right)} \, dS_y - \int\limits_{\infty}^{y} \frac{S_z x}{\sqrt{x^2 + \operatorname{polar\_lift}\left(S_y^2 + S_z^2\right)} \operatorname{polar\_lift}\left(S_y^2\right.}$$