Rotação de Vetores

In [109…

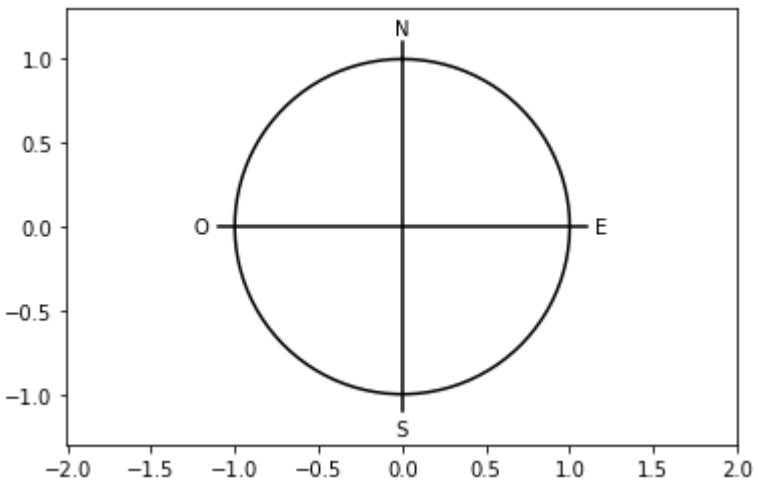
```
%reset
```

In [110…

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import pickle

import rasterio
from rasterio.plot import show

import sys
sys.path.append('c:\guto\gpython\pytools')
from filtbin import filtbin
```

In [111…

```
def desenha_circulo():
    n = np.linspace(0, 2*np.pi, 100)
    circ_x = np.sin(n)
    circ_y = np.cos(n)

    fig, ax = plt.subplots()
    ax.plot(circ_x, circ_y, 'k')
    ax.plot([0, 0], [-1.1, 1.1], 'k')
    ax.plot([-1.1, 1.1],[0, 0], 'k')
    ax.axis('equal')

    ax.text(0, 1.15, 'N', ha='center')
    ax.text(0, -1.15, 'S', ha='center', va='top')
    ax.text(1.15, 0 , 'E', va='center')
    ax.text(-1.15, 0, 'O', ha='right', va='center')
    ax.set_ylim(-1.3, 1.3)

    return fig

desenha_circulo();
```



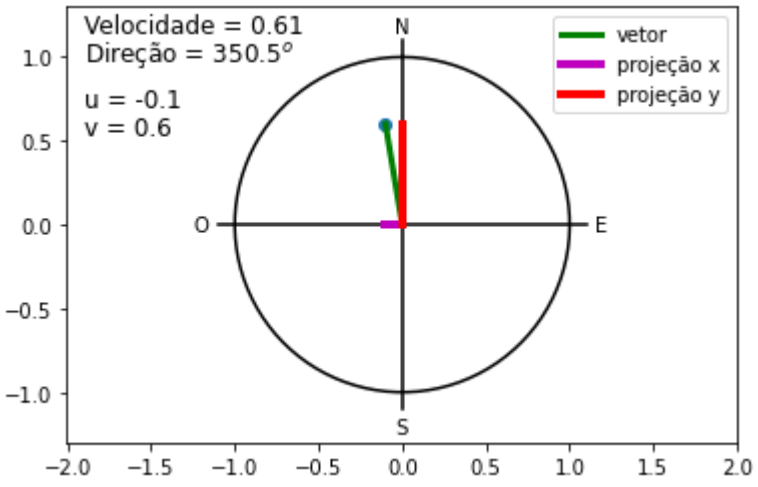
In [ ]:

In [112…

```
def cart2polar(x, y):
    velocidade = (x**2 + y**2)**.5
    theta = np.arctan(y/x) *180/np.pi
    if x > 0:
        theta = 90 - theta
    else:
        theta = 270 - theta
    return velocidade, theta

def polar2cart(vel, direcao):
    u = vel * np.sin(direcao*np.pi/180)
    v = vel * np.cos(direcao*np.pi/180)
    return u, v
```

In [113…

```
valor_x = -.1
valor_y = .6
velocidade, theta = cart2polar(valor_x, valor_y)

u, v = polar2cart(velocidade, theta)

linha_x = [0, valor_x]
linha_y = [0, valor_y]
origem = [0, 0]

desenha_circulo();
plt.plot(valor_x, valor_y, 'o')
plt.plot(linha_x, linha_y, 'g', linewidth=3, label='vetor')
plt.plot(linha_x, origem, 'm', linewidth=4, label='projeção x')
plt.plot(origem, linha_y, 'r', linewidth=4, label='projeção y')
plt.text(-1.9, 1.15, 'Velocidade = '+ str(np.round(velocidade,2)), fontsize=12)
plt.text(-1.9, 0.98, 'Direção = '+ str(np.round(theta,1)) +'$^o$', fontsize=12 )
plt.text(-1.9, 0.70, 'u = '+ str(np.round(u,2)), fontsize=12)
plt.text(-1.9, 0.54, 'v = '+ str(np.round(v,1)), fontsize=12 )
plt.legend()
plt.show()
```



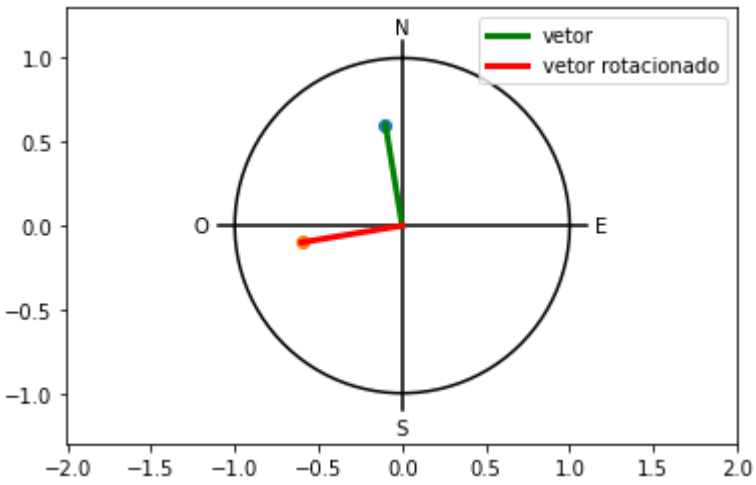
In [114…

```
def rotaciona_vetor(u, v, theta):
    theta_rad = -theta*np.pi/180
    ur = u * np.cos(theta_rad) - v * np.sin(theta_rad)
```

```
    vr = u * np.sin(theta_rad) + v * np.cos(theta_rad)
    return ur, vr
```

In [115…

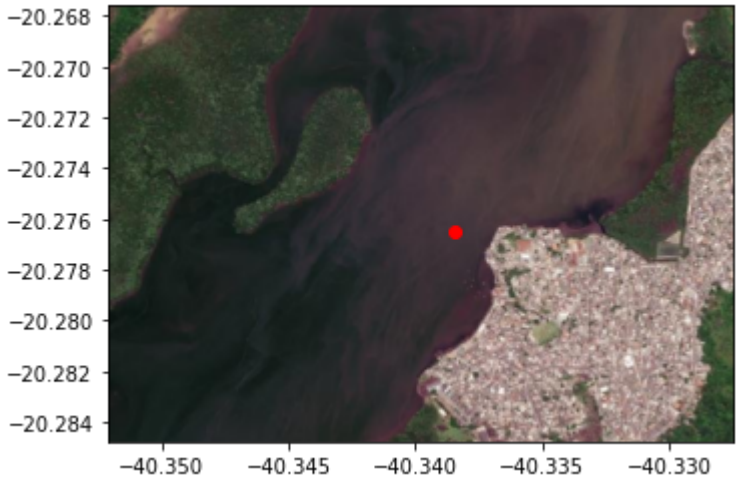
```
valor_x = -.1
valor_y = .6
valor_xr, valor_yr = rotaciona_vetor(valor_x, valor_y, 270)

linha_x = [0, valor_x]
linha_y = [0, valor_y]
linha_x2 = [0, valor_xr]
linha_y2 = [0, valor_yr]

desenha_circulo();
plt.plot(valor_x, valor_y, 'o')
plt.plot(linha_x, linha_y, 'g', linewidth=3, label='vetor')
plt.plot(valor_xr, valor_yr, 'o')
plt.plot(linha_x2, linha_y2, 'r', linewidth=3, label='vetor rotacionado')
plt.legend()
plt.show()
```



In [116…

```
img_file = 'Fundeio_BaiaVitoria_geo.tiff'
img = rasterio.open(img_file)

# posição fundeio
tx = -40.338476
ty = -20.276495

fig, ax = plt.subplots()
ax.plot(tx, ty, 'or')
show(img, ax=ax)
```

Out[116… <AxesSubplot:>



Delinação magnética

https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml

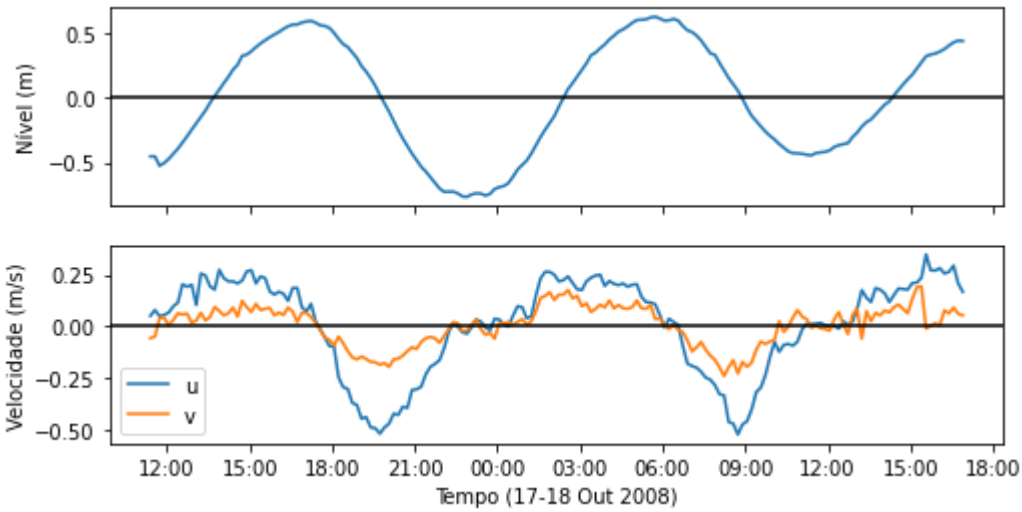
-23.1 graus

In [117…

```
s4_file = 'Tutorial_vetores_dados_S4.pkl'

with open(s4_file, 'rb') as src:
    s4 = pickle.load(src)

tempo = s4[0]
nivel = s4[1] - np.mean(s4[1])
u = s4[2]/100
v = s4[3]/100
```

In [118…

```
fig, (ax1, ax2) = plt.subplots(2,1, figsize=(8,4))

ax1.plot(tempo, nivel)
ax2.plot(tempo, u, label='u')
ax2.plot(tempo, v, label='v')
ax2.legend()
ax1.axhline(color='k')
ax2.axhline(color='k')
ax1.set_ylabel('Nível (m)')
ax2.set_ylabel('Velocidade (m/s)')
ax1.set_xticklabels('')
ax2.xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))
ax2.set_xlabel('Tempo (17-18 Out 2008)')
plt.show()
```

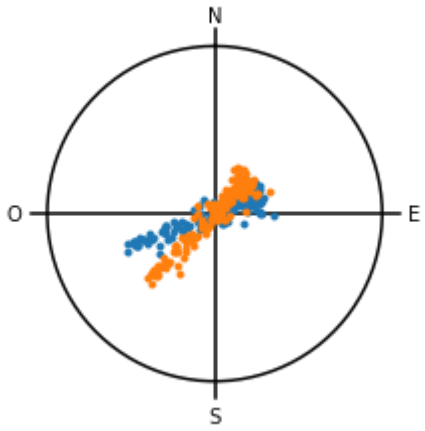


In [119…

```
ur = []
vr = []
for i in range(len(u)):
    cu, cv = rotaciona_vetor(u[i], v[i], -24)
    ur.append(cu)
    vr.append(cv)
```

In [120…

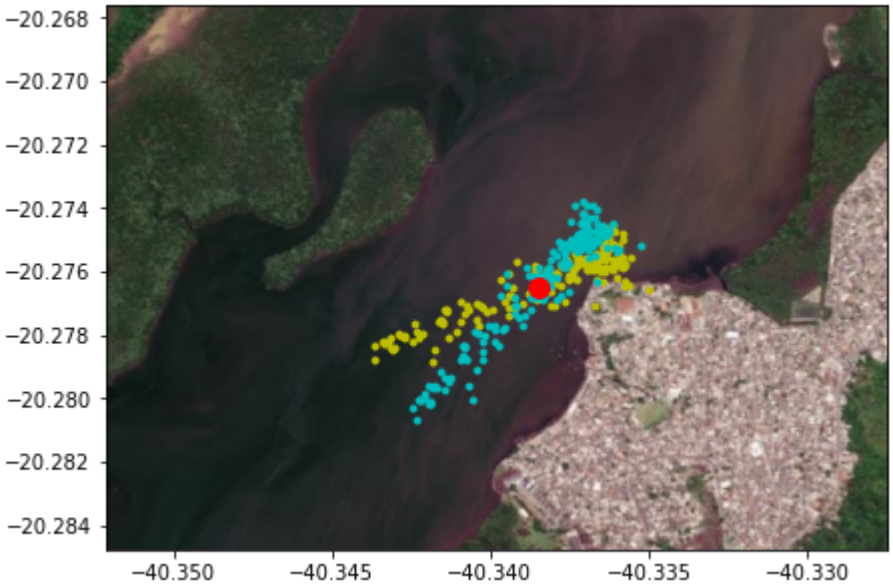
```
desenha_circulo();
plt.plot(u, v, '.')
plt.plot(ur, vr, '.')
plt.axis('off')
plt.show()
```



In [121…

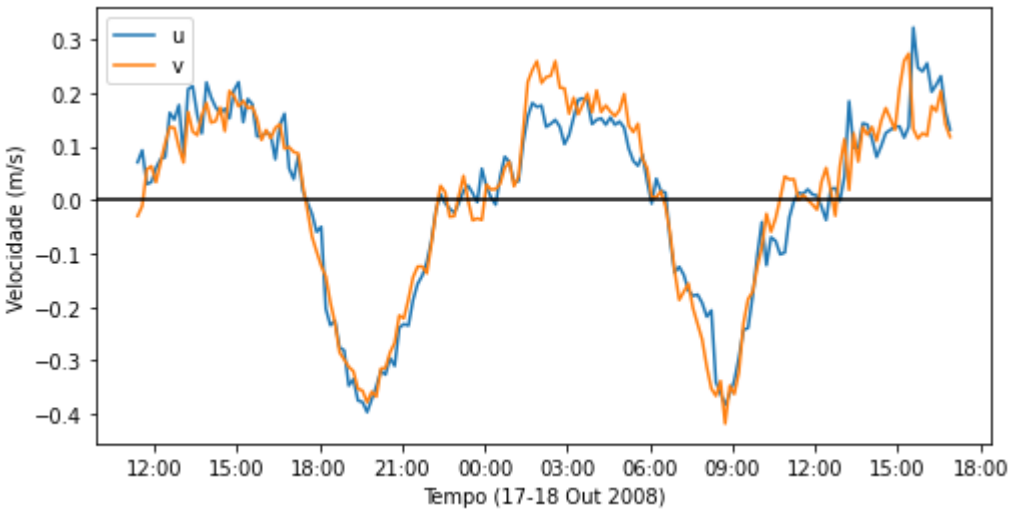
```
ur = []
vr = []
for i in range(len(u)):
    cu, cv = rotaciona_vetor(u[i], v[i], -24)
    ur.append(cu)
    vr.append(cv)

escala = .01
u = np.array(u) * escala + tx
v = np.array(v) * escala + ty
ur2 = np.array(ur) * escala + tx
vr2 = np.array(vr) * escala + ty
```

```
fig, ax = plt.subplots(figsize=(7,7))
show(img, ax=ax)
ax.plot(u, v, 'y.')
ax.plot(ur2, vr2, 'c.')
ax.plot(tx, ty, 'ro', markersize=10)
plt.show()
```



In [122…

```
fig, ax1 = plt.subplots(figsize=(8,4))

ax1.plot(tempo, ur, label='u')
ax1.plot(tempo, vr, label='v')
ax1.legend()
ax1.axhline(color='k')
ax1.set_ylabel('Velocidade (m/s)')
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))
ax1.set_xlabel('Tempo (17-18 Out 2008)')
plt.show()
```

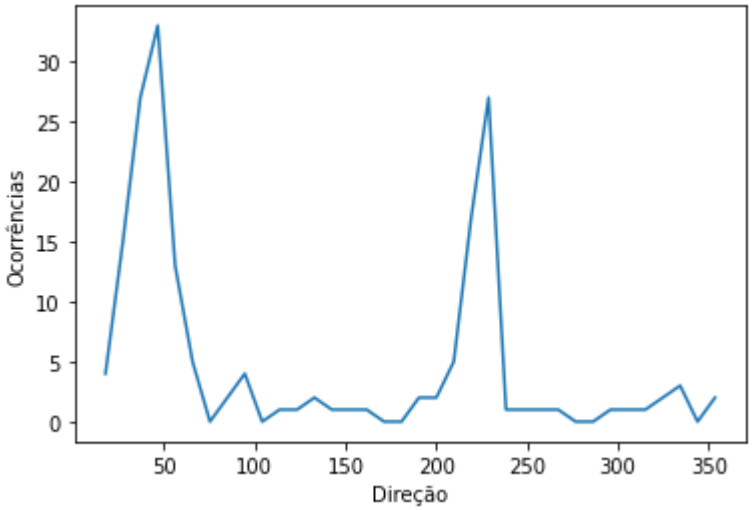


In [123…

```
velocidade = []
direcao = []
for i in range(len(ur)):
```

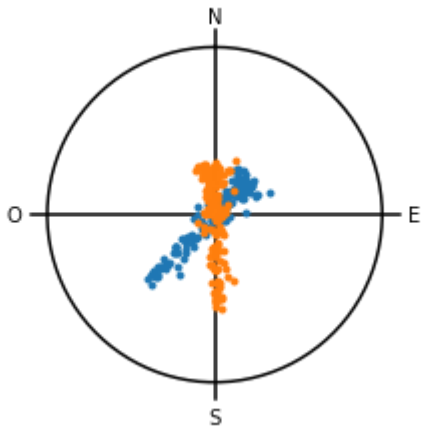
```
    cvel, cdir = cart2polar(ur[i], vr[i])
    velocidade.append(cvel)
    direcao.append(cdir)

a, b = np.histogram(direcao, 36)
b = (b[1:] + b[:-1])/2

plt.plot(b, a)
plt.xlabel('Direção')
plt.ylabel('Ocorrências')

moda = b[a == np.max(a)]
print('moda = ', np.round(moda, 0))
```

```
moda =  [47.]
```



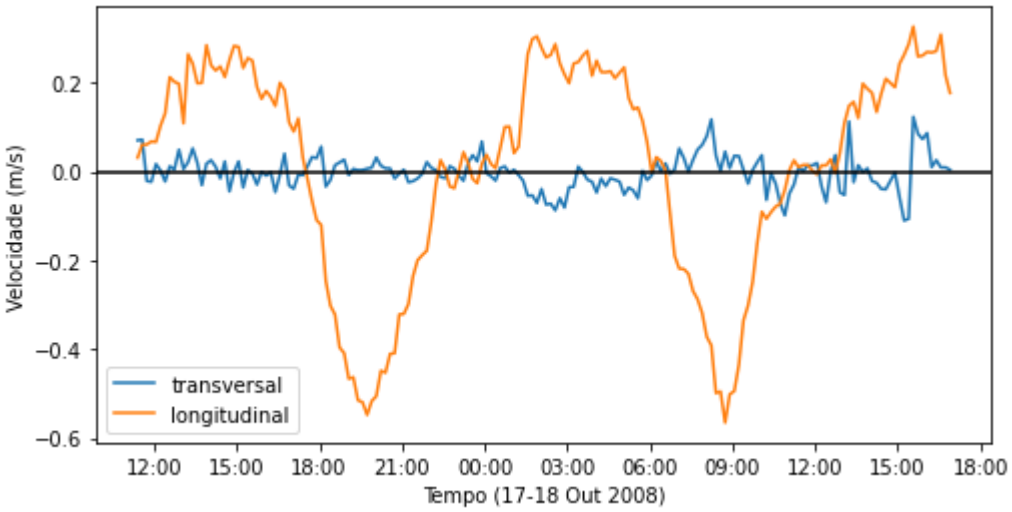
In [124…

```
ur2 = []
vr2 = []
for i in range(len(u)):
    cu, cv = rotaciona_vetor(ur[i], vr[i], -47)
    ur2.append(cu)
    vr2.append(cv)

desenha_circulo();
plt.plot(ur, vr, '.')
plt.plot(ur2, vr2, '.')
plt.axis('off')
plt.show()
```



In [125…

```
fig, ax1 = plt.subplots(figsize=(8,4))

ax1.plot(tempo, ur2, label='transversal')
ax1.plot(tempo, vr2, label='longitudinal')
ax1.legend()
ax1.axhline(color='k')
ax1.set_ylabel('Velocidade (m/s)')
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))
ax1.set_xlabel('Tempo (17-18 Out 2008)')
plt.show()
```



Análise de Componentes Principais

In [126…

```
matriz = np.vstack((ur, vr))

cov_matriz = np.cov(matriz)

auto_valores, auto_vetores = np.linalg.eig(cov_matriz)

variancia_explicada = []
for i in auto_valores:
    variancia_explicada.append((i/sum(auto_valores))*100)

print('formato da matriz de dados =', matriz.shape)
print('variância explicada (%) =', np.round(variancia_explicada, 1))
print('autovalores = ', np.round(auto_valores, 3))
print('autovetores = ', np.round(auto_vetores, 3))
print('matriz de covariancia = ', np.round(cov_matriz, 3))
```

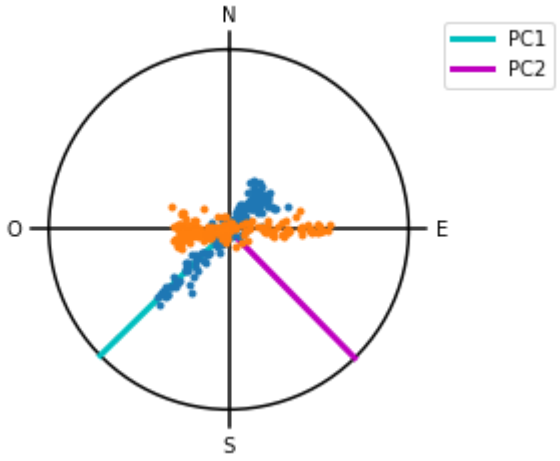
```
formato da matriz de dados = (2, 177)
variância explicada (%) = [ 2.4 97.6]
autovalores =  [0.001 0.058]
autovetores =  [[-0.714 -0.7  ]
 [ 0.7   -0.714]]
matriz de covariancia =  [[0.029 0.028]
 [0.028 0.03 ]]
```

Redução dimensional

https://www.youtube.com/watch?v=ZqXnPcyIAL8

In [129…

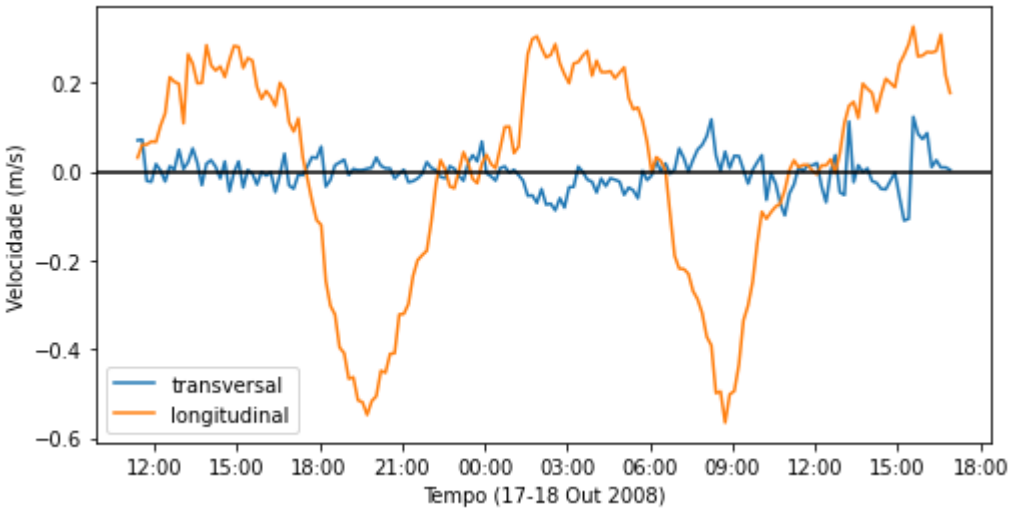
```
ur = np.array(ur)
vr = np.array(vr)

a = auto_vetores[0,0]
b = auto_vetores[0,1]

c = auto_vetores[1,0]
d = auto_vetores[1,1]

ur3 = a*ur + b*vr
vr3 = c*ur + d*vr

desenha_circulo();
plt.plot([0, auto_vetores[0,0]], [0, auto_vetores[0,1]], 'c', linewidth=3, label='PC1')
plt.plot([0, auto_vetores[1,0]], [0, auto_vetores[1,1]], 'm', linewidth=3, label='PC2')

plt.plot(ur, vr, '.')
plt.plot(ur3, vr3, '.')
plt.axis('equal')
plt.axis('off')
plt.legend()
plt.show()
```

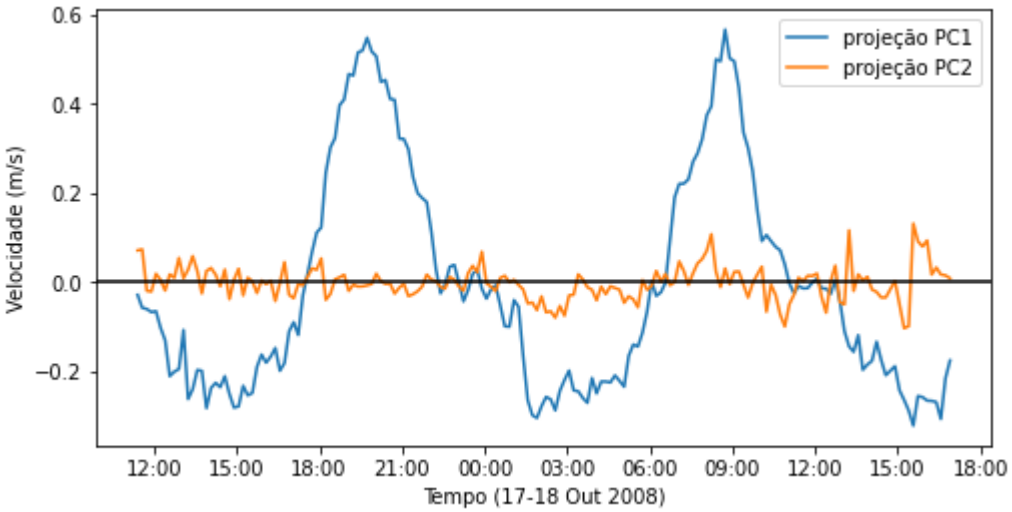


In [130…

```
fig, ax1 = plt.subplots(figsize=(8,4))

ax1.plot(tempo, ur2, label='transversal')
ax1.plot(tempo, vr2, label='longitudinal')
ax1.legend()
ax1.axhline(color='k')
ax1.set_ylabel('Velocidade (m/s)')
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))
ax1.set_xlabel('Tempo (17-18 Out 2008)')
plt.show()

fig, ax1 = plt.subplots(figsize=(8,4))

ax1.plot(tempo, ur3, label='projeção PC1')
ax1.plot(tempo, vr3, label='projeção PC2')
ax1.legend()
ax1.axhline(color='k')
ax1.set_ylabel('Velocidade (m/s)')
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%H:%M'))
```

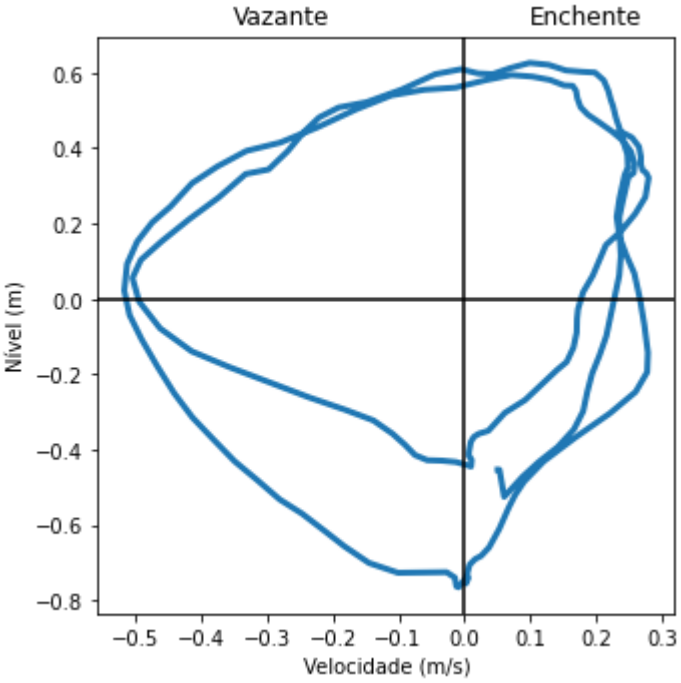
```
ax1.set_xlabel('Tempo (17-18 Out 2008)')
plt.show()
```





In [131…

```
vel_lon = filtbin(ur3, 5) * -1


fig = plt.figure(figsize=(5,5))
ax = fig.add_axes([.1, .1, .8, .8])
ax.plot(vel_lon, nivel, linewidth=3)
ax.axhline(color='k')
ax.axvline(color='k')
ax.set_xlabel('Velocidade (m/s)')
ax.set_ylabel('Nível (m)')
ax.text(-.35, .73, 'Vazante', fontsize=12)
ax.text(.1, .73, 'Enchente', fontsize=12)
plt.show()
```


Vazante
Enchente
Nível (m)
Velocidade (m/s)
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−0.5
−0.4
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3