Dataset_Z_Exploration

March 15, 2021

0.1 Dataset Prueba 1 - Tesis Javier-Uriel

0.1.1 Importamos algunas librerías que nos serán útiles más adelante

[1]:
```
import os
import time
import random

import pandas as pd # for dataframe operations.
import numpy as np #for linear algebra operations.
import seaborn as sns # data visualization library
import matplotlib.pyplot as plt # for plotting

from scipy.fftpack import fft, fftfreq

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import pacf

pd.set_option('display.max_columns', None) #Para mostrar todas las columnas
random.seed(1)
```

0.1.2 Leemos el Dataset

[2]:
```
#Dataset solo movimientos en Z
rpm_list = ['RPM0', 'RPM1', 'RPM2', 'RPM3']
# states_list_org = ["vz","az", "uvz",
#                    "p", "q",
#                    "wp", "wq",
#                    "ap", "aq"]
states_list_org = ["vz","az", "uvz"]
states_list_min = ["vz","az", "uvz"]
dataset_name = "Dataset_Z6_Disturbance"
directory = "../logs/Datasets/"+dataset_name
ORDER = 3
dfs = []
states_list=states_list_org.copy()
```

0.1.3 Corregir la salida

El estado que entrega Pybullet de RPMs es la salida anterior, en este dataset se tomará RPMs como la salida actual. Si el primer elemento de RPMs es 0, es necesario hacer el shift

[3]:
```
for filename in os.listdir(directory):
    if not filename.endswith(".csv"):
        continue
    df = pd.read_csv(os.path.join(directory, filename))
    if any(df['z']<=1): #Eliminar si el dron se cae
        print(filename)
    else:
        if any(df[rpm_list].loc[0]==0): #Desplazar los estados de RPM si es␣
 ↪necesario
            df[rpm_list] = df[rpm_list].shift(periods=-1)
            df = df.dropna()
            df.to_csv(os.path.join(directory, filename), index=False)

        a = []
        ## Desplazamos estados anteriores
        for n in range(1,ORDER+1):
            for column in states_list:
                df[column+str(n)] = df[column].shift(periods=n, fill_value=0)
                a.append(column+str(n))
        dfs.append(df)
states_list+=a

dataset = pd.concat(dfs)
dataset.describe()
```

[3]:

	timestamps	x	y	z	Q1	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	4.996731e+01	8.335401e-02	-9.044891e-02	5.319935e+01	-1.075346e-04	
std	2.885594e+01	6.987598e-02	7.093276e-02	1.037135e+01	1.145220e-02	
min	0.000000e+00	-1.414761e-01	-2.280038e+00	3.385300e+01	-4.761323e-01	
25%	2.497917e+01	4.262207e-02	-1.292551e-01	4.997121e+01	-9.933642e-08	
50%	4.996250e+01	7.618597e-02	-8.226449e-02	5.016746e+01	0.000000e+00	
75%	7.494167e+01	1.183252e-01	-4.790182e-02	5.203965e+01	9.397464e-08	
max	9.999167e+01	4.973182e+00	7.583118e-01	1.779624e+02	9.211059e-01	

	Q2	Q3	Q4	p	q	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	-1.206161e-04	5.768941e-05	9.998517e-01	-3.214013e-04	-2.703013e-04	
std	8.816835e-03	4.282112e-03	8.323757e-03	2.651956e-02	1.837012e-02	
min	-1.995036e-01	-6.931737e-01	-4.999858e-01	-3.137499e+00	-9.770365e-01	
25%	-8.942428e-08	-1.165184e-05	9.999999e-01	-2.141099e-07	-1.953285e-07	
50%	0.000000e+00	-7.814737e-06	1.000000e+00	0.000000e+00	-0.000000e+00	
75%	8.931078e-08	-2.150338e-07	1.000000e+00	2.067380e-07	1.979739e-07	

```
max    2.254375e-01  6.932850e-01  1.000000e+00  3.137265e+00  3.047570e-01

                r            vx            vy            vz            wp  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean   3.131701e-05  1.192967e-03 -1.219761e-03  5.355558e-02  1.884070e-04
std    8.728354e-03  3.765168e-02  2.942748e-02  7.678060e-01  1.252994e-01
min   -1.616839e+00 -4.926684e-01 -2.271822e+00 -1.431533e+01 -3.672953e+00
25%   -2.332544e-05 -2.096181e-07 -2.976484e-07 -1.174247e-01 -1.832466e-06
50%   -1.565858e-05 -5.081633e-17 -1.308150e-16  2.425708e-17  4.264982e-17
75%   -4.532093e-07  2.630532e-07  2.235253e-07  1.831781e-01  1.814993e-06
max    2.225088e-01  9.571342e+00  3.154290e+00  9.395058e+00  5.732395e+00

               wq            wr            ax            ay            az  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean  -1.247069e-04  2.739298e-05  3.614468e-04  1.418952e-05 -1.099231e-04
std    1.230224e-01  1.470166e-02  1.460567e-01  1.420266e-01  1.683938e+00
min   -2.720019e+00 -1.282142e+00 -3.517441e+00 -1.112667e+01 -9.800000e+00
25%   -1.714102e-06 -2.586595e-08 -1.780675e-06 -2.102573e-06 -6.665799e-03
50%    3.360138e-17  1.267736e-07  7.662684e-18 -5.651989e-18  7.593925e-13
75%    1.630613e-06  1.992890e-07  1.924635e-06  1.916839e-06  2.093719e-02
max    2.546305e+00  1.006867e+00  1.939510e+01  7.799504e+00  1.608245e+01

                ap            aq            ar          RPM0          RPM1  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean   6.251896e-05 -4.036304e-05  1.206084e-05  1.442266e+04  1.442251e+04
std    2.111760e+00  2.101450e+00  6.914199e-01  1.244395e+03  1.246140e+03
min   -1.706654e+01 -1.911974e+01 -1.843849e+01  9.440300e+03  9.440300e+03
25%   -1.996062e-05 -1.809994e-05 -2.263765e-09  1.442173e+04  1.441921e+04
50%   -1.423747e-18 -3.679440e-25 -1.034803e-09  1.446843e+04  1.446843e+04
75%    1.811841e-05  1.707612e-05  1.945982e-09  1.455289e+04  1.455384e+04
max    1.703098e+01  1.810688e+01  2.294702e+01  2.166645e+04  2.166645e+04

              RPM2          RPM3         ux         uy         uz         uvx  \
count  6.355284e+06  6.355284e+06  6355284.0  6355284.0  6355284.0  6355284.0
mean   1.442267e+04  1.442258e+04        0.0        0.0       50.0        0.0
std    1.244217e+03  1.245455e+03        0.0        0.0        0.0        0.0
min    9.440300e+03  9.440300e+03        0.0        0.0       50.0        0.0
25%    1.442171e+04  1.442039e+04        0.0        0.0       50.0        0.0
50%    1.446843e+04  1.446843e+04        0.0        0.0       50.0        0.0
75%    1.455308e+04  1.455413e+04        0.0        0.0       50.0        0.0
max    2.166645e+04  2.166645e+04        0.0        0.0       50.0        0.0

             uvy           uvz         up         uq         ur         uwp  \
count  6355284.0  6.355284e+06  6355284.0  6355284.0  6355284.0  6355284.0
mean         0.0  4.164795e-02        0.0        0.0        0.0        0.0
std          0.0  7.938309e-01        0.0        0.0        0.0        0.0
min          0.0 -9.606008e+00        0.0        0.0        0.0        0.0
```

```
25%          0.0 -1.314332e-01        0.0        0.0        0.0        0.0
50%          0.0  0.000000e+00        0.0        0.0        0.0        0.0
75%          0.0  1.789063e-01        0.0        0.0        0.0        0.0
max          0.0  9.606008e+00        0.0        0.0        0.0        0.0

              uwq        uwr           vz1           az1          uvz1  \
count   6355284.0  6355284.0  6.355284e+06  6.355284e+06  6.355284e+06
mean          0.0        0.0  5.355604e-02 -1.115869e-04  4.164710e-02
std           0.0        0.0  7.677797e-01  1.683889e+00  7.938161e-01
min           0.0        0.0 -1.431533e+01 -9.800000e+00 -9.606008e+00
25%           0.0        0.0 -1.174084e-01 -6.662233e-03 -1.314201e-01
50%           0.0        0.0  2.404872e-17  7.527312e-13  0.000000e+00
75%           0.0        0.0  1.831473e-01  2.092986e-02  1.788774e-01
max           0.0        0.0  9.395058e+00  1.608245e+01  9.606008e+00

                vz2           az2           uvz2           vz3           az3  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean   5.355651e-02 -1.132458e-04  4.164625e-02  5.355698e-02 -1.148901e-04
std    7.677532e-01  1.683840e+00  7.938013e-01  7.677267e-01  1.683792e+00
min   -1.431533e+01 -9.800000e+00 -9.606008e+00 -1.431533e+01 -9.800000e+00
25%   -1.173794e-01 -6.658932e-03 -1.314015e-01 -1.173556e-01 -6.655904e-03
50%    2.380960e-17  7.460699e-13  0.000000e+00  2.367857e-17  7.460699e-13
75%    1.831147e-01  2.092228e-02  1.788774e-01  1.830731e-01  2.091375e-02
max    9.395058e+00  1.608245e+01  9.606008e+00  9.395058e+00  1.608245e+01

               uvz3
count  6.355284e+06
mean   4.164539e-02
std    7.937866e-01
min   -9.606008e+00
25%   -1.313499e-01
50%    0.000000e+00
75%    1.788647e-01
max    9.606008e+00
```

0.1.4 Estados repetidos

En este caso se eliminan estados repetidos y estados que se encuentren en estado transitorio mientras el dron despega o se estabiliza antes de introducir la señal de control.

[4]:
```
shape_b4 = dataset.drop(["timestamps"], axis=1).shape
shape_drop= dataset.drop(["timestamps"], axis=1).drop_duplicates().shape
print(f'shape (b4 drop) = {shape_b4}')
print(f'shape = {shape_drop}')
print(f'len (b4 drop) - len = {shape_b4[0]-shape_drop[0]}')
```

```
shape (b4 drop) = (6355284, 47)
shape = (6330453, 47)
```

```
len (b4 drop) - len = 24831
```

[5]:
```
states = dataset.drop(["timestamps"], axis=1).drop_duplicates()[states_list]
print(f'columns = {states.columns}')
print(f'shape = {states.shape}')
states.head()
```

```
columns = Index(['vz', 'az', 'uvz', 'vz1', 'az1', 'uvz1', 'vz2', 'az2', 'uvz2',
'vz3',
       'az3', 'uvz3'],
      dtype='object')
shape = (6330453, 12)
```

[5]:

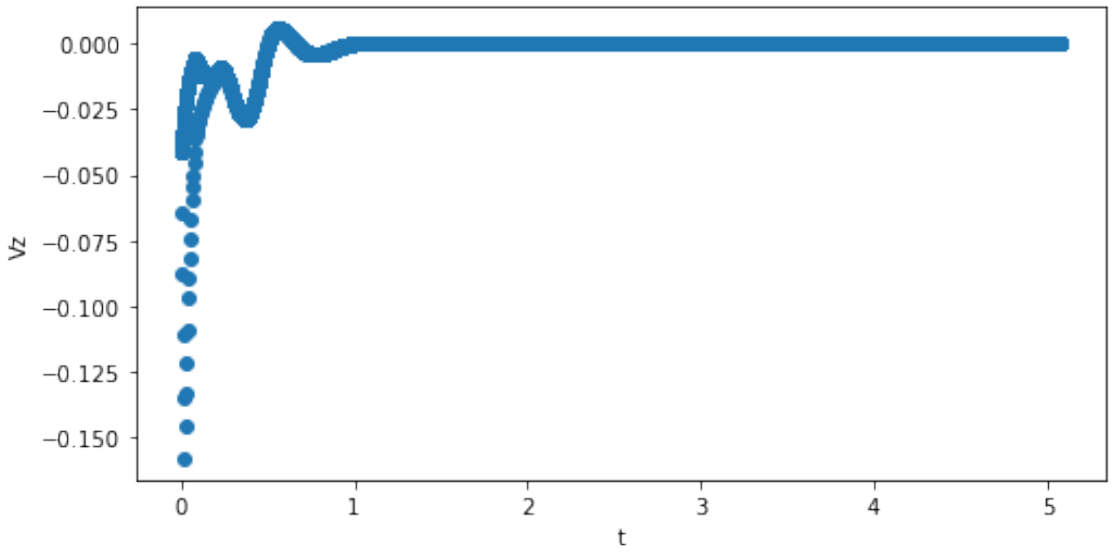
	vz	az	uvz	vz1	az1	uvz1	vz2	\
0	-0.040833	-9.800000	0.809266	0.000000	0.000000	0.000000	0.000000	
1	0.009909	12.178210	0.809266	-0.040833	-9.800000	0.809266	0.000000	
2	0.060643	12.176110	0.809266	0.009909	12.178210	0.809266	-0.040833	
3	0.111368	12.173937	0.809266	0.060643	12.176110	0.809266	0.009909	
4	0.162083	12.171559	0.809266	0.111368	12.173937	0.809266	0.060643	

	az2	uvz2	vz3	az3	uvz3
0	0.00000	0.000000	0.000000	0.00000	0.000000
1	0.00000	0.000000	0.000000	0.00000	0.000000
2	-9.80000	0.809266	0.000000	0.00000	0.000000
3	12.17821	0.809266	-0.040833	-9.80000	0.809266
4	12.17611	0.809266	0.009909	12.17821	0.809266

[6]:
```
states_duplicates = dataset[dataset.duplicated(keep='last')]
states_duplicates = states_duplicates.dropna()
```

[7]:
```
fig = plt.figure(figsize=(8, 4))
t = states_duplicates['timestamps']
y = states_duplicates['vz']
#y_ref = states_duplicates['uvz']
plt.scatter(t, y)
#plt.scatter(t, y_ref)
plt.ylabel('Vz')
plt.xlabel('t')
```

[7]:
```
Text(0.5, 0, 't')
```



[8]:
```
#Eliminar del dataset los estados repetidos entre 20 y 25 segundos
# for filename in os.listdir(directory):
#     if not filename.endswith(".csv"):
#         continue
#     df = pd.read_csv(os.path.join(directory, filename))
#     df = df[(df['timestamps']>20) & (df['timestamps']<25)]
#     df = pd.concat([df, states_duplicates])
#     df = df.reset_index(drop=True)
#     df_gpby = df.groupby(list(df.columns))
#     idx = [x[0] for x in df_gpby.groups.values() if len(x) > 1]
#     if len(idx)>1:
#         print(filename)
```

[9]:
```
df = pd.read_csv(os.path.join(directory, filename))
df = df[df['timestamps']>5]
df.head()
```

[9]:

	timestamps	x	y	z	Q1	Q2	\
1201	5.004167	0.06663	-0.066273	52.195232	5.214055e-08	5.143486e-08	
1202	5.008333	0.06663	-0.066273	52.195312	5.043469e-08	4.974850e-08	
1203	5.012500	0.06663	-0.066273	52.195385	4.846667e-08	4.780551e-08	
1204	5.016667	0.06663	-0.066273	52.195451	4.623652e-08	4.560592e-08	
1205	5.020833	0.06663	-0.066273	52.195509	4.374429e-08	4.314978e-08	

	Q3	Q4	p	q	r	vx	\
1201	-0.000012	1.0	1.042799e-07	1.028710e-07	-0.000024	3.087617e-08	
1202	-0.000012	1.0	1.008682e-07	9.949822e-08	-0.000024	3.500213e-08	
1203	-0.000012	1.0	9.693218e-08	9.561218e-08	-0.000024	3.901568e-08	

```
1204 -0.000012  1.0  9.247194e-08  9.121295e-08 -0.000024  4.289677e-08
1205 -0.000012  1.0  8.748754e-08  8.630062e-08 -0.000024  4.662537e-08

                vy        vz            wp            wq            wr  \
1201 -3.099948e-08  0.021037 -6.929587e-07 -6.862388e-07  1.029498e-07
1202 -3.518176e-08  0.019274 -8.188213e-07 -8.094428e-07  1.095894e-07
1203 -3.925052e-08  0.017511 -9.446630e-07 -9.326262e-07  1.162278e-07
1204 -4.318529e-08  0.015749 -1.070484e-06 -1.055789e-06  1.228652e-07
1205 -4.696561e-08  0.013987 -1.196283e-06 -1.178932e-06  1.295014e-07

                ax            ay        az       ap       aq        ar  \
1201  1.012401e-06 -1.026078e-06 -0.423153 -0.00003 -0.00003  0.000002
1202  9.902309e-07 -1.003747e-06 -0.423080 -0.00003 -0.00003  0.000002
1203  9.632507e-07 -9.765024e-07 -0.423006 -0.00003 -0.00003  0.000002
1204  9.314616e-07 -9.443446e-07 -0.422933 -0.00003 -0.00003  0.000002
1205  8.948651e-07 -9.072756e-07 -0.422860 -0.00003 -0.00003  0.000002

              RPM0          RPM1          RPM2          RPM3   ux   uy    uz  \
1201  14153.320770  14153.318786  14153.320819  14153.323311  0.0  0.0  50.0
1202  14153.320770  14153.318786  14153.320819  14153.323311  0.0  0.0  50.0
1203  14153.320770  14153.318786  14153.320819  14153.323311  0.0  0.0  50.0
1204  14153.320770  14153.318786  14153.320819  14153.323311  0.0  0.0  50.0
1205  14275.941456  14275.941862  14275.941446  14275.941045  0.0  0.0  50.0

      uvx  uvy  uvz   up   uq   ur  uwp  uwq  uwr
1201  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
1202  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
1203  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
1204  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
1205  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

0.1.5 Se grafican los datos

Se grafica un histograma de cada una de las propiedades los datos analizados individualmente por columnas, en el cual se observa que todos tienen distribuciones altamente apuntadas (curosis) y en algunos casos bimodales, pero de cualquier manera, no son uniformes

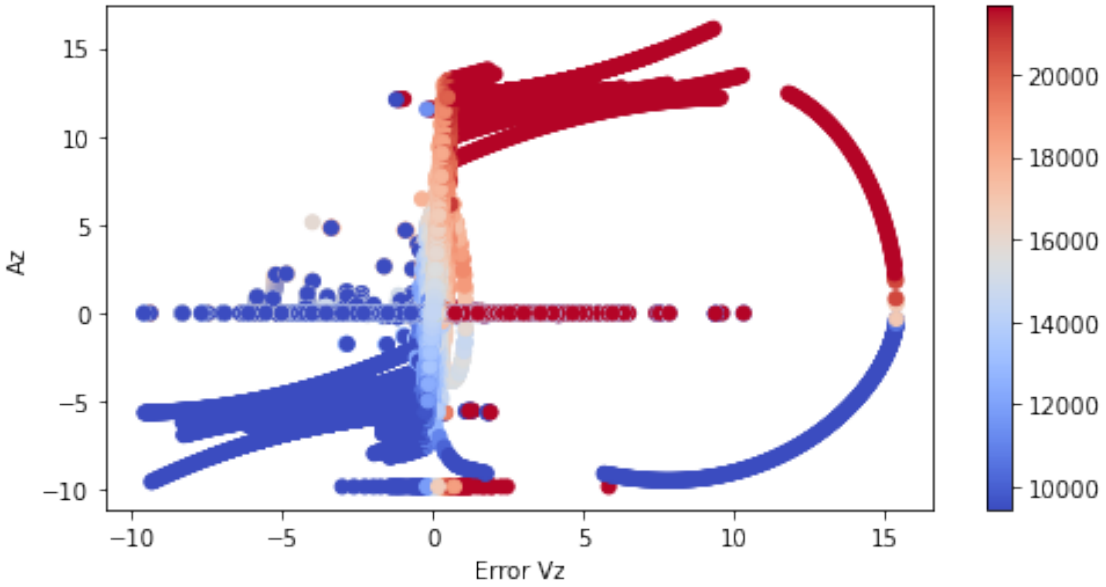
[10]:
```
n_bins = 50
#_ = dataset.hist(bins=n_bins, figsize=(30,30))
```

0.1.6 Análisis de estados

[11]:
```
fig = plt.figure(figsize=(8, 4))
x = dataset['uvz']-dataset['vz']
y = dataset['az']
c = dataset['RPM0']
plt.scatter(x, y, c=c, cmap='coolwarm')
```

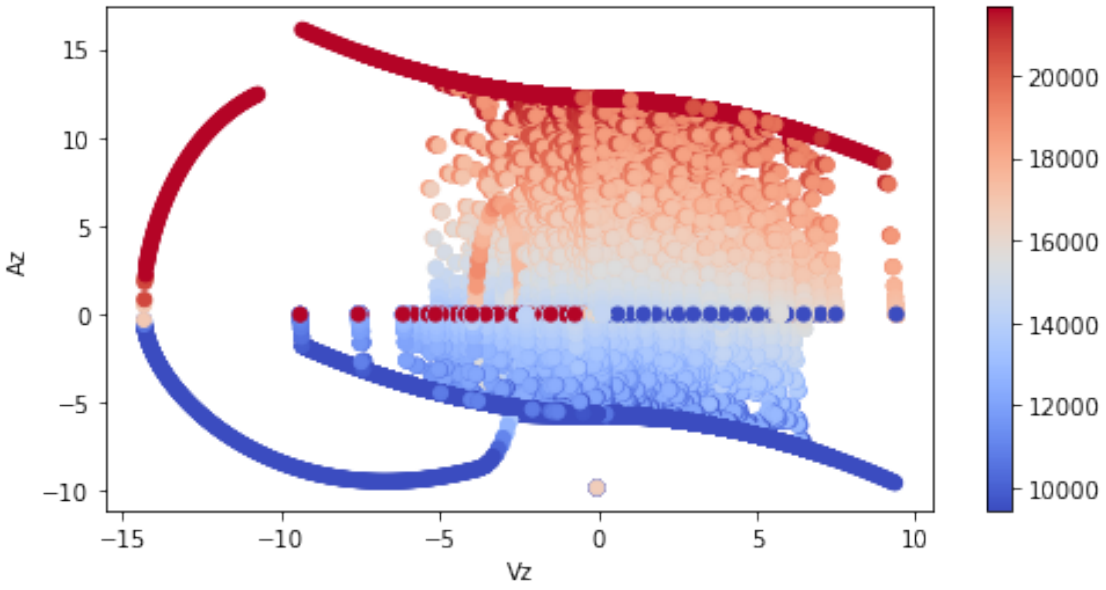
```
plt.colorbar()
plt.ylabel('Az')
plt.xlabel('Error Vz')
```

[11]: Text(0.5, 0, 'Error Vz')



[12]:
```
fig = plt.figure(figsize=(8, 4))
x = dataset['vz']
plt.scatter(x, y, c=c, cmap='coolwarm')
plt.colorbar()
plt.ylabel('Az')
plt.xlabel('Vz')
```

[12]: Text(0.5, 0, 'Vz')



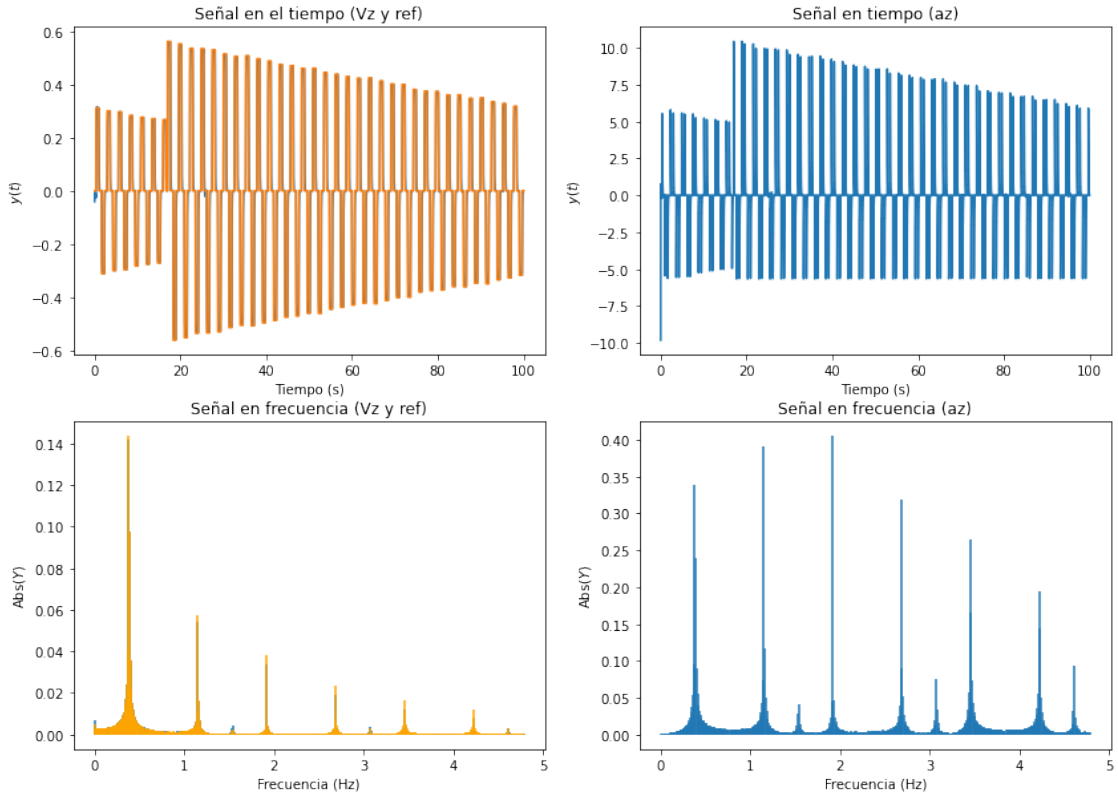
0.1.7 Análisis de Fourier

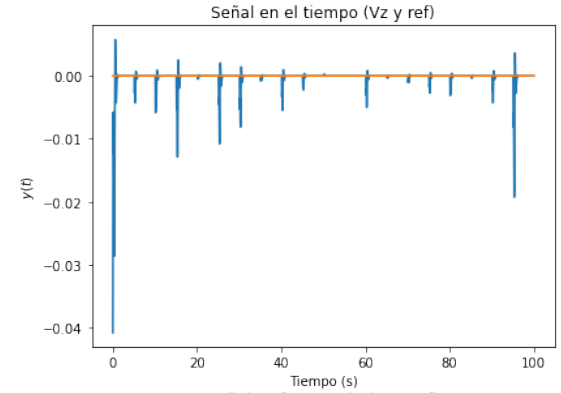
Gráfica de algunas señales

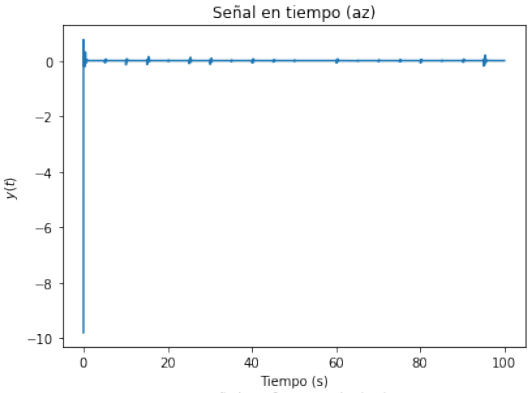
[13]:
```
def plot_fourier(df, states=['vz', 'uvz', 'az']):
    dt = df['timestamps'][1]-df['timestamps'][0]
    n = len(df['timestamps'])
    Y = fft(df[states[0]].to_numpy()) / n # Transformada normalizada
    Y_ref = fft(df[states[1]].to_numpy()) / n
    frq = fftfreq(n, dt)
    fig = plt.figure(figsize=(14, 10))
    ax1 = fig.add_subplot(221)
    ax1.plot(df['timestamps'], df[states[0]], df['timestamps'], df[states[1]])
    ax1.set_xlabel('Tiempo (s)')
    ax1.set_ylabel('$y(t)$')
    ax1.set_title('Señal en el tiempo (Vz y ref)')
    ax2 = fig.add_subplot(223)
    ax2.set_title('Señal en frecuencia (Vz y ref)')
    ax2.vlines(frq[0:int(n/50)], 0, abs(Y[0:int(n/50)]))
    ax2.vlines(frq[0:int(n/50)], 0, abs(Y_ref[0:int(n/50)]),  color='orange')
    plt.xlabel('Frecuencia (Hz)')
    plt.ylabel('Abs($Y$)')

    Y = fft(df[states[2]].to_numpy()) / n # Transformada normalizada
    ax1 = fig.add_subplot(222)
    ax1.plot(df['timestamps'], df[states[2]])
    ax1.set_xlabel('Tiempo (s)')
    ax1.set_ylabel('$y(t)$')
```

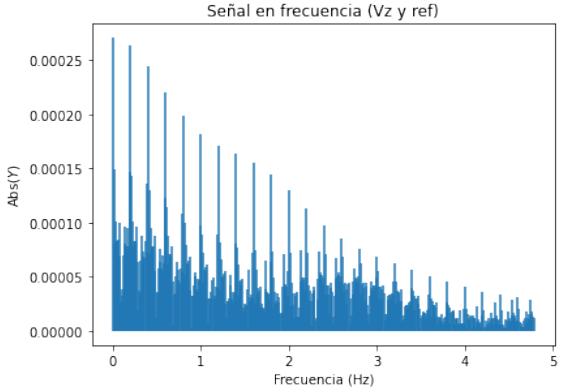
```
    ax1.set_title('Señal en tiempo (az)')
    ax2 = fig.add_subplot(224)
    ax2.set_title('Señal en frecuencia (az)')
    ax2.vlines(frq[0:int(n/50)], 0, abs(Y[0:int(n/50)]))
    plt.xlabel('Frecuencia (Hz)')
    plt.ylabel('Abs($Y$)')
```

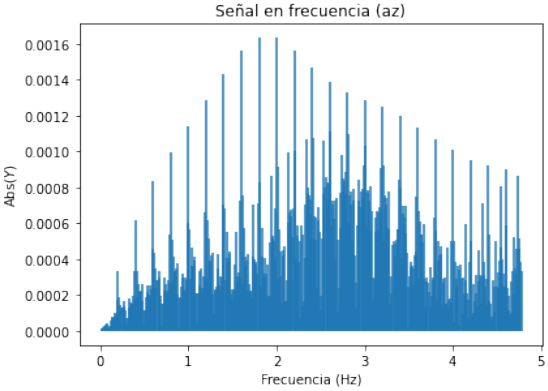
[14]:
```
for df in random.choices(dfs, k = 8):
    plot_fourier(df)
```

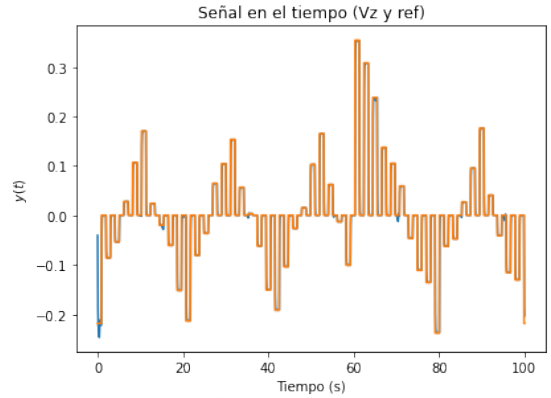


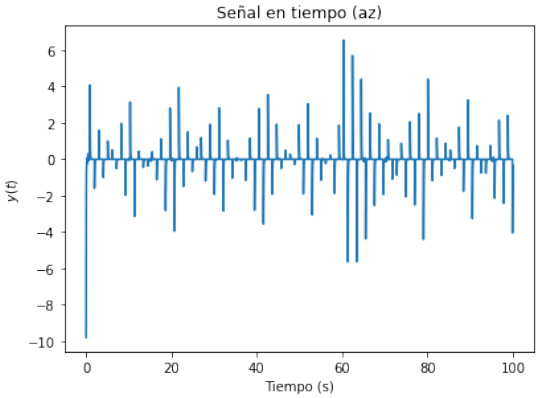

Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)

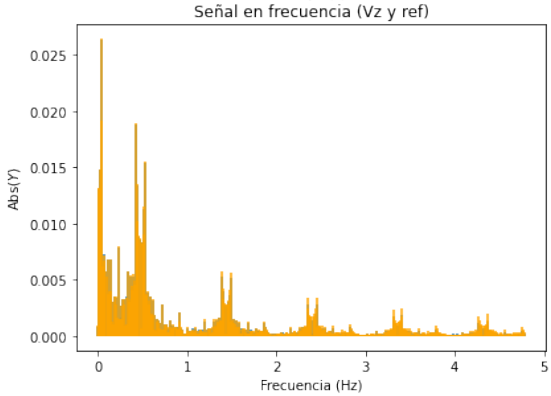

Señal en tiempo (az)
y(t)
Tiempo (s)

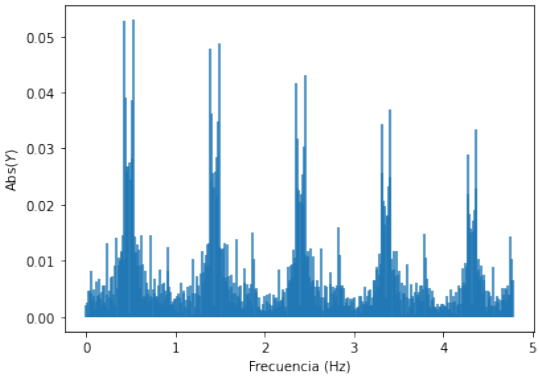

Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)

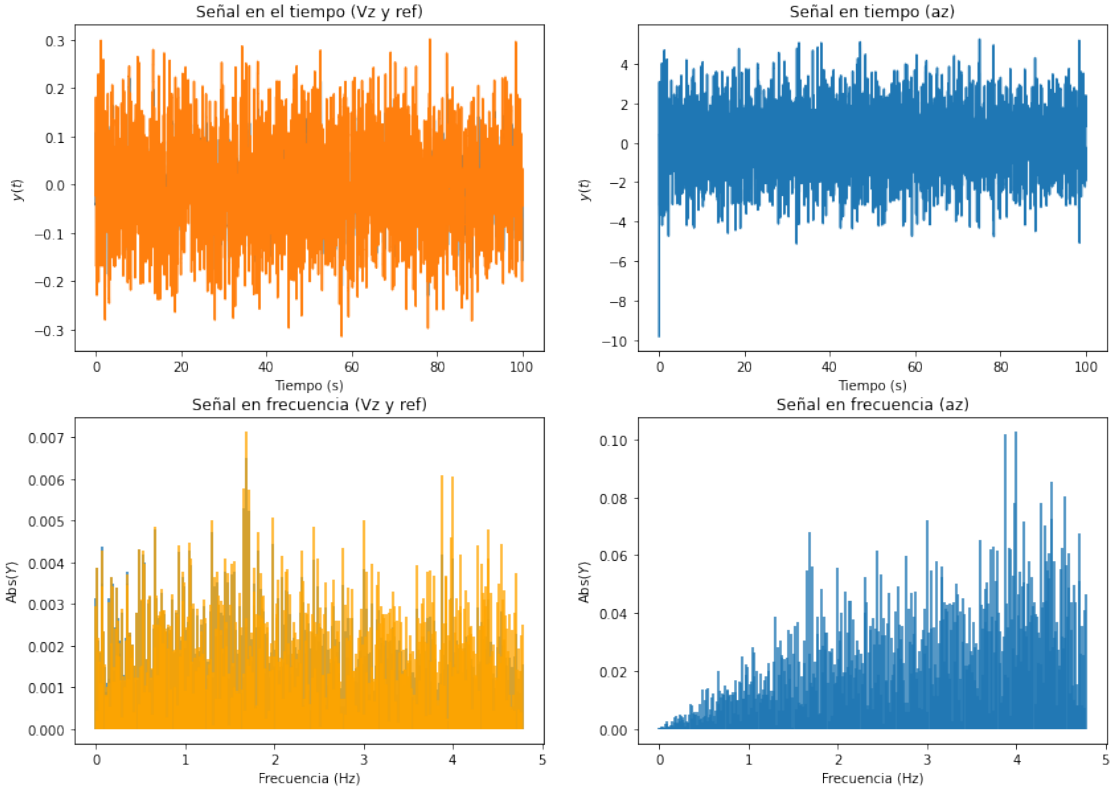

Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)

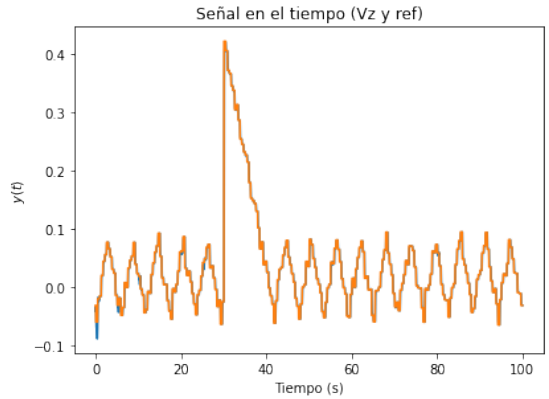

Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)

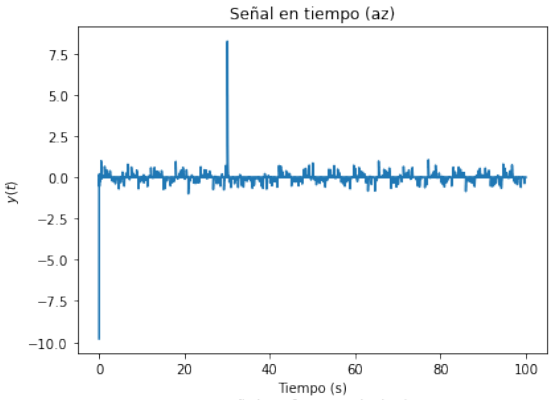

Señal en tiempo (az)
y(t)
Tiempo (s)

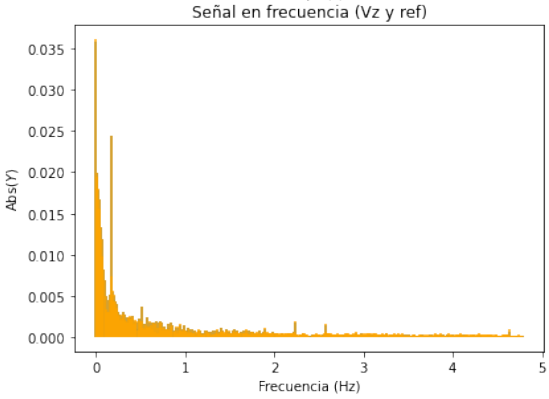

Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)

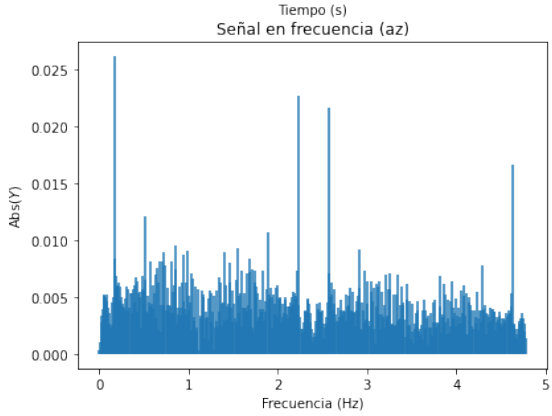

Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)

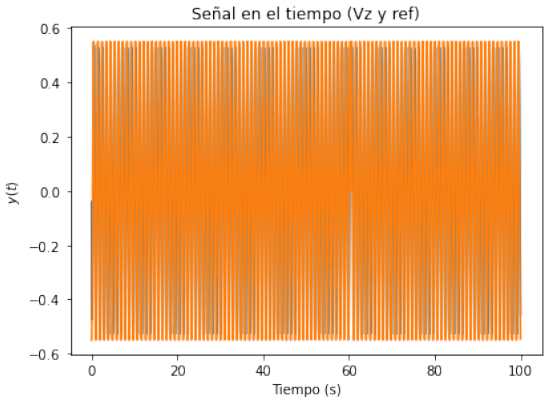

Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)
Señal en tiempo (az)
y(t)
Tiempo (s)
Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)
Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)

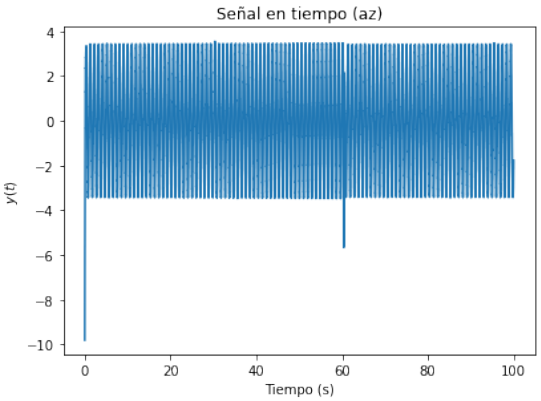

Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)

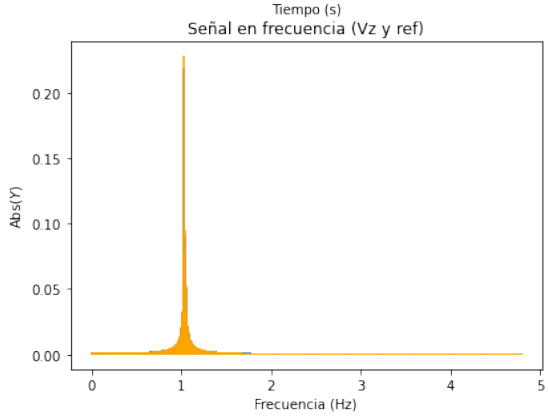

Señal en tiempo (az)
y(t)
Tiempo (s)

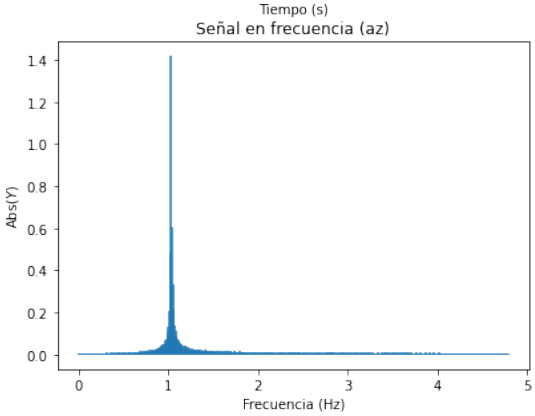

Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)

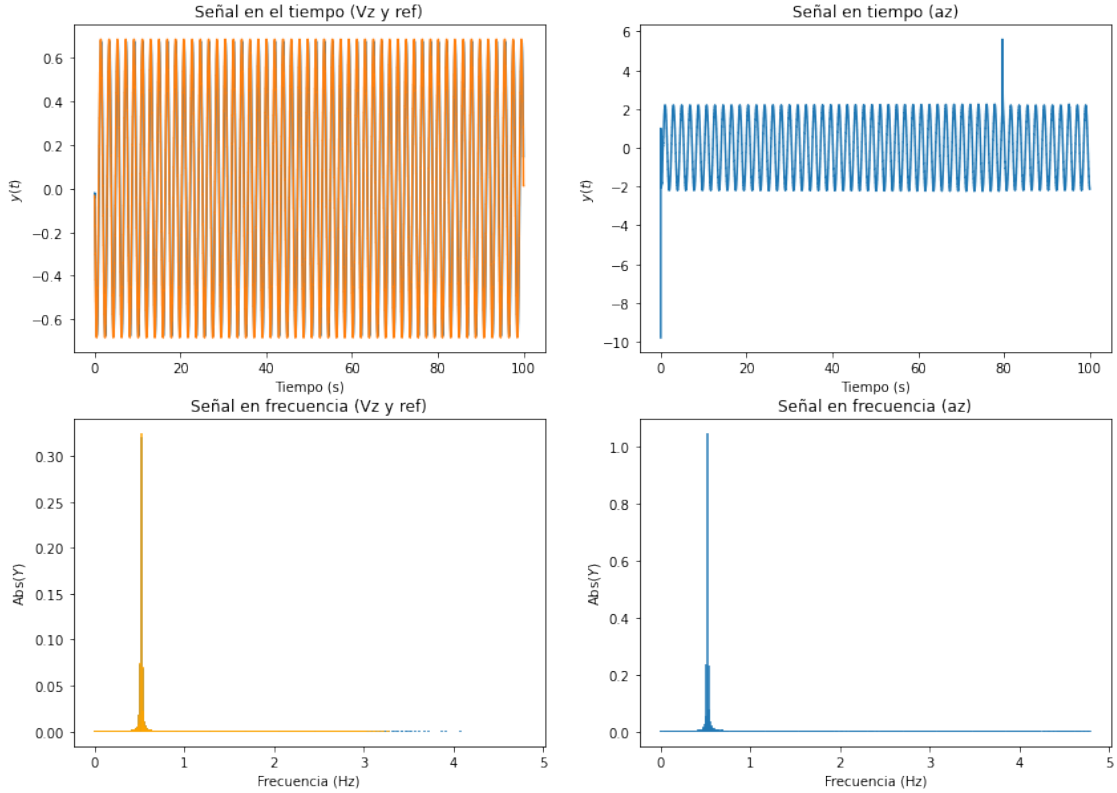

Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)

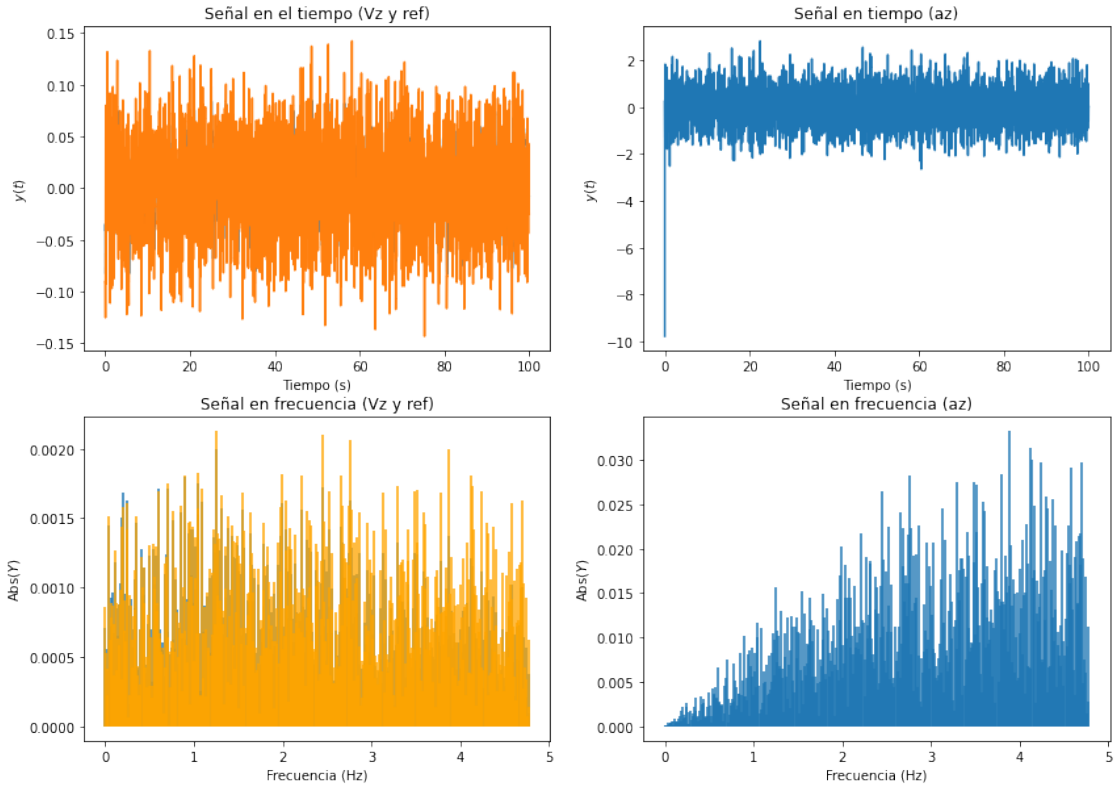

Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)


Señal en tiempo (az)
y(t)
Tiempo (s)


Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)


Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)


Señal en el tiempo (Vz y ref)
y(t)
Tiempo (s)
Señal en tiempo (az)
y(t)
Tiempo (s)
Señal en frecuencia (Vz y ref)
Abs(Y)
Frecuencia (Hz)
Señal en frecuencia (az)
Abs(Y)
Frecuencia (Hz)



Histograma

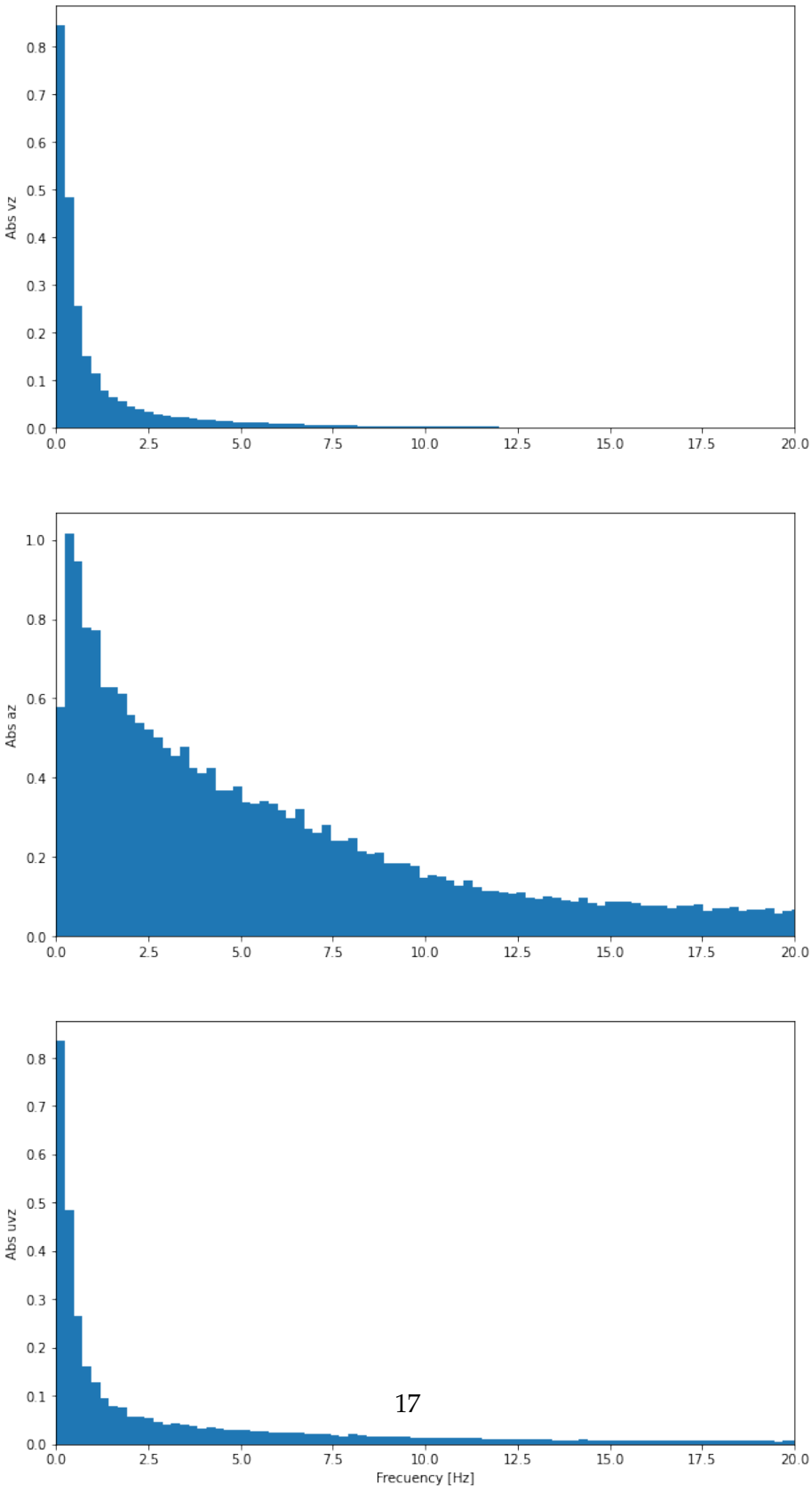
[15]:
```
Fourier = []
for i, df in enumerate(dfs):
    dt = df['timestamps'][1]-df['timestamps'][0]
    n = len(df['timestamps'])
    Fourier.append({})
    for state in states_list_org:
        Fourier[i][state]={}
        Fourier[i][state]['Y'] = abs(fft(df[state].to_numpy())/n)[0:int(n/2)] #␣
 ↪Transformada normalizada
        Fourier[i][state]['X'] = fftfreq(n, dt)[0:int(n/2)]
```

[16]:
```
F = {}
for state in states_list_org:
    F[state]={}
    F[state]['X'] = []
    F[state]['Y'] = []
    for f in Fourier:
        F[state]['X'] = np.concatenate([F[state]['X'], f[state]['X']])
        F[state]['Y'] = np.concatenate([F[state]['Y'], f[state]['Y']])
```

[17]:
```
fig, axs = plt.subplots(len(states_list_org), 1, figsize=(10, 20))
fig.suptitle('Fourier Transform Histogram per State')
for i, state in enumerate(states_list_org):
    axs[i].hist(F[state]['X'], bins=10*n_bins, weights=((F[state]['Y']+1e-7)/
 ↪len(Fourier)))
    axs[i].set_ylabel(f'Abs {state}')
    axs[i].set_xlim(0, 20)
axs[i].set_xlabel('Frecuency [Hz]')
```

[17]:
```
Text(0.5, 0, 'Frecuency [Hz]')
```

Fourier Transform Histogram per State



0.1.8 Análisis de Características - Método Estático

[18]:
```
dataset.describe()
```

[18]:

	timestamps	x	y	z	Q1	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	4.996731e+01	8.335401e-02	-9.044891e-02	5.319935e+01	-1.075346e-04	
std	2.885594e+01	6.987598e-02	7.093276e-02	1.037135e+01	1.145220e-02	
min	0.000000e+00	-1.414761e-01	-2.280038e+00	3.385300e+01	-4.761323e-01	
25%	2.497917e+01	4.262207e-02	-1.292551e-01	4.997121e+01	-9.933642e-08	
50%	4.996250e+01	7.618597e-02	-8.226449e-02	5.016746e+01	0.000000e+00	
75%	7.494167e+01	1.183252e-01	-4.790182e-02	5.203965e+01	9.397464e-08	
max	9.999167e+01	4.973182e+00	7.583118e-01	1.779624e+02	9.211059e-01	

	Q2	Q3	Q4	p	q	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	-1.206161e-04	5.768941e-05	9.998517e-01	-3.214013e-04	-2.703013e-04	
std	8.816835e-03	4.282112e-03	8.323757e-03	2.651956e-02	1.837012e-02	
min	-1.995036e-01	-6.931737e-01	-4.999858e-01	-3.137499e+00	-9.770365e-01	
25%	-8.942428e-08	-1.165184e-05	9.999999e-01	-2.141099e-07	-1.953285e-07	
50%	0.000000e+00	-7.814737e-06	1.000000e+00	0.000000e+00	-0.000000e+00	
75%	8.931078e-08	-2.150338e-07	1.000000e+00	2.067380e-07	1.979739e-07	
max	2.254375e-01	6.932850e-01	1.000000e+00	3.137265e+00	3.047570e-01	

	r	vx	vy	vz	wp	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	3.131701e-05	1.192967e-03	-1.219761e-03	5.355558e-02	1.884070e-04	
std	8.728354e-03	3.765168e-02	2.942748e-02	7.678060e-01	1.252994e-01	
min	-1.616839e+00	-4.926684e-01	-2.271822e+00	-1.431533e+01	-3.672953e+00	
25%	-2.332544e-05	-2.096181e-07	-2.976484e-07	-1.174247e-01	-1.832466e-06	
50%	-1.565858e-05	-5.081633e-17	-1.308150e-16	2.425708e-17	4.264982e-17	
75%	-4.532093e-07	2.630532e-07	2.235253e-07	1.831781e-01	1.814993e-06	
max	2.225088e-01	9.571342e+00	3.154290e+00	9.395058e+00	5.732395e+00	

	wq	wr	ax	ay	az	\
count	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	6.355284e+06	
mean	-1.247069e-04	2.739298e-05	3.614468e-04	1.418952e-05	-1.099231e-04	
std	1.230224e-01	1.470166e-02	1.460567e-01	1.420266e-01	1.683938e+00	
min	-2.720019e+00	-1.282142e+00	-3.517441e+00	-1.112667e+01	-9.800000e+00	
25%	-1.714102e-06	-2.586595e-08	-1.780675e-06	-2.102573e-06	-6.665799e-03	
50%	3.360138e-17	1.267736e-07	7.662684e-18	-5.651989e-18	7.593925e-13	
75%	1.630613e-06	1.992890e-07	1.924635e-06	1.916839e-06	2.093719e-02	
max	2.546305e+00	1.006867e+00	1.939510e+01	7.799504e+00	1.608245e+01	

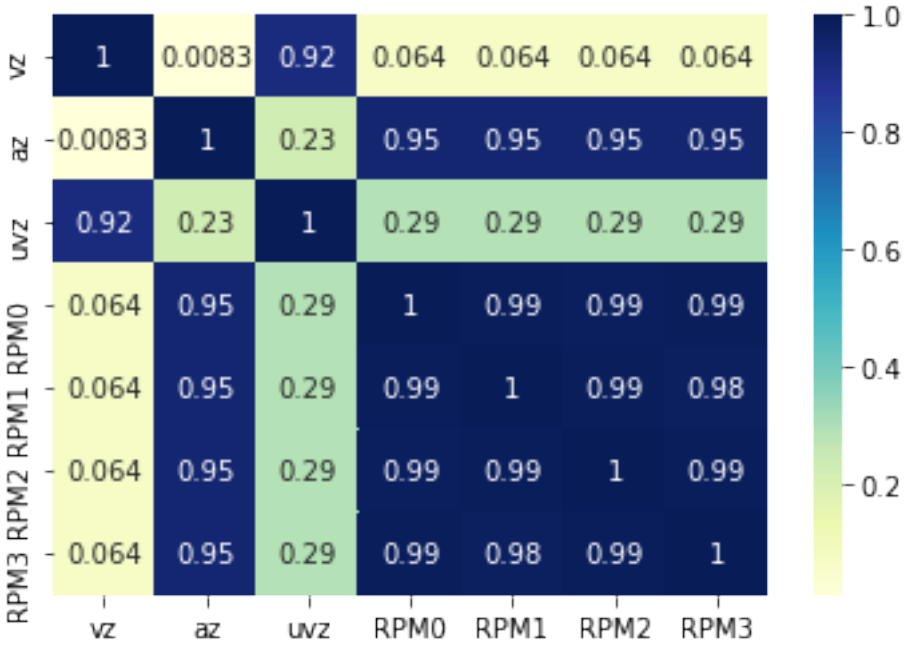
```
                ap            aq            ar          RPM0          RPM1  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean   6.251896e-05 -4.036304e-05  1.206084e-05  1.442266e+04  1.442251e+04
std    2.111760e+00  2.101450e+00  6.914199e-01  1.244395e+03  1.246140e+03
min   -1.706654e+01 -1.911974e+01 -1.843849e+01  9.440300e+03  9.440300e+03
25%   -1.996062e-05 -1.809994e-05 -2.263765e-09  1.442173e+04  1.441921e+04
50%   -1.423747e-18 -3.679440e-25 -1.034803e-09  1.446843e+04  1.446843e+04
75%    1.811841e-05  1.707612e-05  1.945982e-09  1.455289e+04  1.455384e+04
max    1.703098e+01  1.810688e+01  2.294702e+01  2.166645e+04  2.166645e+04

               RPM2          RPM3         ux         uy         uz        uvx  \
count  6.355284e+06  6.355284e+06  6355284.0  6355284.0  6355284.0  6355284.0
mean   1.442267e+04  1.442258e+04        0.0        0.0       50.0        0.0
std    1.244217e+03  1.245455e+03        0.0        0.0        0.0        0.0
min    9.440300e+03  9.440300e+03        0.0        0.0       50.0        0.0
25%    1.442171e+04  1.442039e+04        0.0        0.0       50.0        0.0
50%    1.446843e+04  1.446843e+04        0.0        0.0       50.0        0.0
75%    1.455308e+04  1.455413e+04        0.0        0.0       50.0        0.0
max    2.166645e+04  2.166645e+04        0.0        0.0       50.0        0.0

            uvy           uvz         up         uq         ur         uwp  \
count  6355284.0  6.355284e+06  6355284.0  6355284.0  6355284.0  6355284.0
mean         0.0  4.164795e-02        0.0        0.0        0.0        0.0
std          0.0  7.938309e-01        0.0        0.0        0.0        0.0
min          0.0 -9.606008e+00        0.0        0.0        0.0        0.0
25%          0.0 -1.314332e-01        0.0        0.0        0.0        0.0
50%          0.0  0.000000e+00        0.0        0.0        0.0        0.0
75%          0.0  1.789063e-01        0.0        0.0        0.0        0.0
max          0.0  9.606008e+00        0.0        0.0        0.0        0.0

            uwq        uwr           vz1           az1           uvz1  \
count  6355284.0  6355284.0  6.355284e+06  6.355284e+06  6.355284e+06
mean         0.0        0.0  5.355604e-02 -1.115869e-04  4.164710e-02
std          0.0        0.0  7.677797e-01  1.683889e+00  7.938161e-01
min          0.0        0.0 -1.431533e+01 -9.800000e+00 -9.606008e+00
25%          0.0        0.0 -1.174084e-01 -6.662233e-03 -1.314201e-01
50%          0.0        0.0  2.404872e-17  7.527312e-13  0.000000e+00
75%          0.0        0.0  1.831473e-01  2.092986e-02  1.788774e-01
max          0.0        0.0  9.395058e+00  1.608245e+01  9.606008e+00

              vz2           az2           uvz2          vz3           az3  \
count  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06  6.355284e+06
mean   5.355651e-02 -1.132458e-04  4.164625e-02  5.355698e-02 -1.148901e-04
std    7.677532e-01  1.683840e+00  7.938013e-01  7.677267e-01  1.683792e+00
min   -1.431533e+01 -9.800000e+00 -9.606008e+00 -1.431533e+01 -9.800000e+00
25%   -1.173794e-01 -6.658932e-03 -1.314015e-01 -1.173556e-01 -6.655904e-03
50%    2.380960e-17  7.460699e-13  0.000000e+00  2.367857e-17  7.460699e-13
```

```
75%     1.831147e-01  2.092228e-02  1.788774e-01  1.830731e-01  2.091375e-02
max     9.395058e+00  1.608245e+01  9.606008e+00  9.395058e+00  1.608245e+01
```

```
               uvz3
count  6.355284e+06
mean   4.164539e-02
std    7.937866e-01
min   -9.606008e+00
25%   -1.313499e-01
50%    0.000000e+00
75%    1.788647e-01
max    9.606008e+00
```

Mapa de Correlación

[19]:
```
correlation = dataset[states_list_org + rpm_list].corr() #corr() method of␣
 ↪pandas library calculates correlation between columns of dataframe
sns.heatmap(correlation,cmap="YlGnBu",annot=True)
plt.show()
```



Análisis de Correlaciones

[20]:
```
# Comentado porque se demora mucho procesando
# for i in states_list_org:
#     sns.lmplot(x=i, y=rpm_list[0], data=dataset,line_kws={'color': 'red'})
#     text="Relation between RPM0 and " + i
```

```
#     plt.title(text)
#     plt.show()
```

[21]:
```
corr_df = pd.DataFrame()
for i in rpm_list:
    correlation = dataset.corr()[i] # convert series to dataframe so it can be␣
 ↪sorted
    correlation_df = pd.DataFrame(correlation) # correct column label from␣
 ↪Points to correlation
    correlation_df.columns = [f"Correlation_{i}"] # sort correlation
    corr_df = pd.concat([correlation_df, corr_df], axis=1)
corr_df = corr_df.dropna(how='all')
corr_df = corr_df.sort_values(by=[f'Correlation_{rpm_list[0]}'], ascending=False)
corr_df.head(30)
```

[21]:

	Correlation_RPM3	Correlation_RPM2	Correlation_RPM1	\
RPM0	0.988489	0.987050	0.988441	
RPM3	1.000000	0.988602	0.981782	
RPM1	0.981782	0.988386	1.000000	
RPM2	0.988602	1.000000	0.988386	
az	0.951401	0.952748	0.951559	
az1	0.924830	0.926165	0.925057	
az2	0.898248	0.899570	0.898543	
az3	0.871656	0.872965	0.872017	
uvz	0.290540	0.290807	0.290390	
uvz1	0.288956	0.289223	0.288808	
uvz2	0.287372	0.287640	0.287227	
uvz3	0.285789	0.286056	0.285646	
vz	0.063534	0.063695	0.063540	
vz1	0.054842	0.054990	0.054846	
ap	-0.063903	-0.044897	0.070915	
vz2	0.046392	0.046528	0.046394	
vz3	0.038185	0.038309	0.038185	
q	0.049384	-0.014283	-0.035367	
Q2	0.049521	-0.017002	-0.038974	
ay	-0.035700	-0.016665	0.046328	
ax	0.045023	-0.014410	-0.032766	
Q4	0.014395	0.014965	0.013625	
vx	0.010417	0.012402	0.011664	
vy	0.013145	0.011484	0.012058	
x	0.006824	0.008132	0.008905	
wr	0.004877	0.000470	-0.020765	
wq	0.003270	-0.004294	-0.001963	
y	-0.000102	-0.000554	-0.002291	
timestamps	-0.003574	-0.003726	-0.003492	
r	-0.007101	-0.003398	-0.007013	

```
            Correlation_RPM0
RPM0                1.000000
RPM3                0.988489
RPM1                0.988441
RPM2                0.987050
az                  0.952743
az1                 0.926136
az2                 0.899518
az3                 0.872889
uvz                 0.290831
uvz1                0.289246
uvz2                0.287660
uvz3                0.286075
vz                  0.063655
vz1                 0.054950
ap                  0.049748
vz2                 0.046489
vz3                 0.038270
q                   0.029738
Q2                  0.028904
ay                  0.027671
ax                  0.026145
Q4                  0.014465
vx                  0.011632
vy                  0.011228
x                   0.007356
wr                  0.001287
wq                 -0.000992
y                  -0.001694
timestamps         -0.003691
r                  -0.003936
```

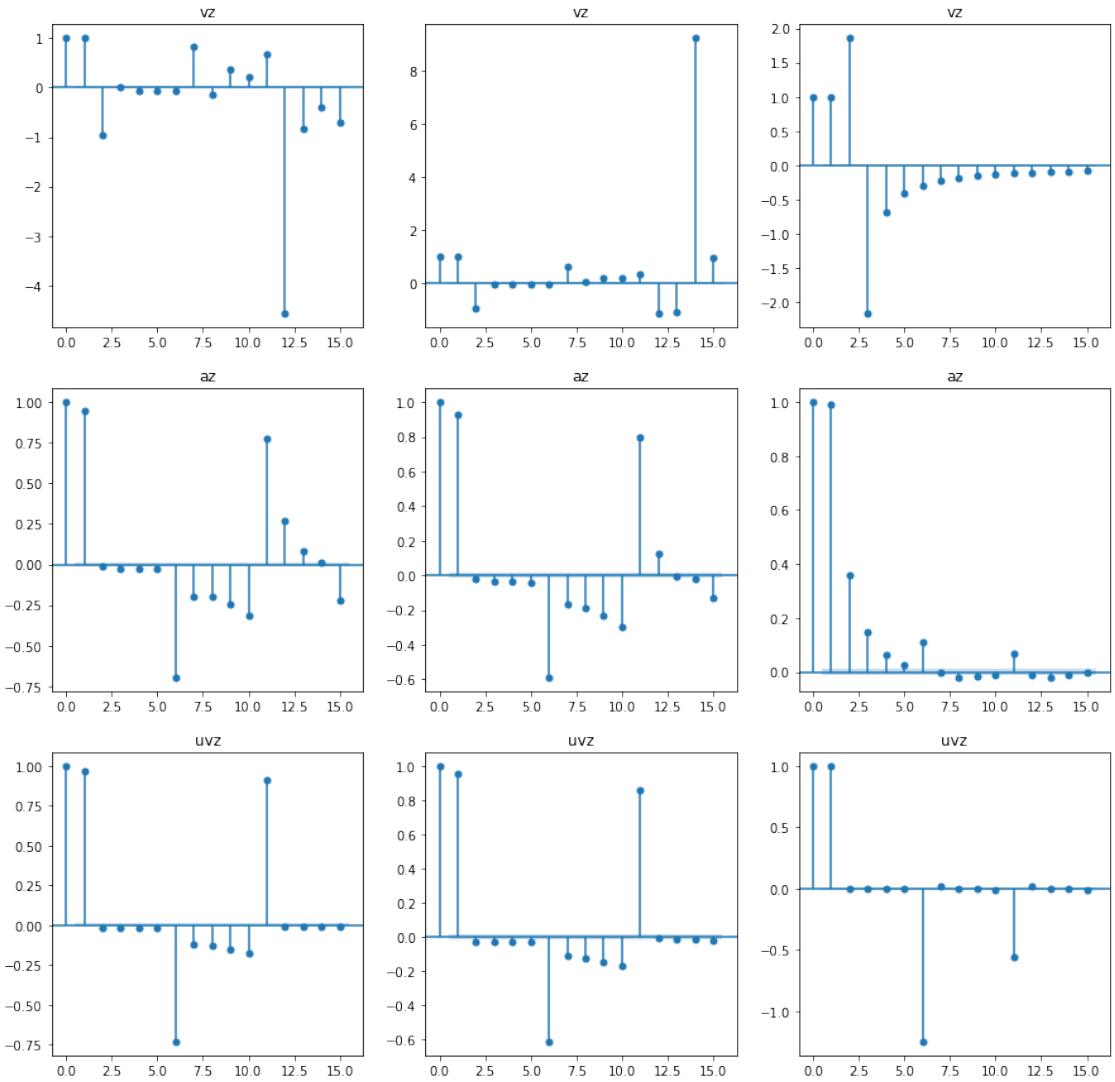
0.1.9 Análisis de Características - Método Dinámico

Autocorrelación Parcial

[22]:
```
N_df = 3
nlags = 15
fig, axs = plt.subplots(N_df, len(states_list_min), figsize=(15, 15))
for k, df in enumerate(random.choices(dfs, k = N_df)):
    for j, i in enumerate(states_list_min):
        plot_pacf(df[i], lags=nlags, ax = axs[j, k])
        axs[j, k].set_title(i)
```

```
C:\Users\mrjar\.conda\envs\tesis\lib\site-
packages\statsmodels\regression\linear_model.py:1434: RuntimeWarning: invalid
value encountered in sqrt
  return rho, np.sqrt(sigmasq)
C:\Users\mrjar\.conda\envs\tesis\lib\site-
```

```
packages\statsmodels\regression\linear_model.py:1434: RuntimeWarning: invalid
value encountered in sqrt
  return rho, np.sqrt(sigmasq)
C:\Users\mrjar\.conda\envs\tesis\lib\site-
packages\statsmodels\regression\linear_model.py:1434: RuntimeWarning: invalid
value encountered in sqrt
  return rho, np.sqrt(sigmasq)
C:\Users\mrjar\.conda\envs\tesis\lib\site-
packages\statsmodels\regression\linear_model.py:1434: RuntimeWarning: invalid
value encountered in sqrt
  return rho, np.sqrt(sigmasq)
```



[23]:
```
pacf_df = [pd.DataFrame()]*len(states_list_min)
for k, df in enumerate(dfs):
```

```
    for j, i in enumerate(states_list_min):
        tmp = pd.DataFrame(pacf(df[i], nlags=nlags), columns=[str(k)])
        pacf_df[j] = pd.concat([pacf_df[j], tmp], axis=1)
pacf_df_dict = {}
for j, i in enumerate(states_list_min):
    pacf_df_dict[i] =  pd.DataFrame()
    pacf_df_dict[i]['mean'] = pacf_df[j].T.mean()
    pacf_df_dict[i]['min'] = pacf_df[j].T.min()
    pacf_df_dict[i]['max'] = pacf_df[j].T.max()
    pacf_df_dict[i]['abs'] = np.maximum(pacf_df[j].T.max(), abs(pacf_df[j].T.
 ↪min()))
```

```
C:\Users\mrjar\.conda\envs\tesis\lib\site-
packages\statsmodels\regression\linear_model.py:1434: RuntimeWarning: invalid
value encountered in sqrt
  return rho, np.sqrt(sigmasq)
```

```
---------------------------------------------------------------------------
LinAlgError                               Traceback (most recent call last)
<ipython-input-23-0431c926cf39> in <module>
      2 for k, df in enumerate(dfs):
      3     for j, i in enumerate(states_list_min):
----> 4         tmp = pd.DataFrame(pacf(df[i], nlags=nlags), columns=[str(k)])
      5         pacf_df[j] = pd.concat([pacf_df[j], tmp], axis=1)
      6 pacf_df_dict = {}

~\.conda\envs\tesis\lib\site-packages\statsmodels\tsa\stattools.py in pacf(x,␣
 ↪nlags, method, alpha)
   1042         ret = pacf_ols(x, nlags=nlags, efficient=efficient,␣
 ↪adjusted=adjusted)
   1043     elif method in ("yw", "ywa", "ywadjusted", "yw_adjusted"):
-> 1044         ret = pacf_yw(x, nlags=nlags, method="adjusted")
   1045     elif method in ("ywm", "ywmle", "yw_mle"):
   1046         ret = pacf_yw(x, nlags=nlags, method="mle")

~\.conda\envs\tesis\lib\site-packages\statsmodels\tsa\stattools.py in pacf_yw(x,␣
 ↪nlags, method)
    754     pacf = [1.0]
    755     for k in range(1, nlags + 1):
--> 756         pacf.append(yule_walker(x, k, method=method)[0][-1])
    757     return np.array(pacf)
    758

~\.conda\envs\tesis\lib\site-packages\statsmodels\regression\linear_model.py in␣
 ↪yule_walker(x, order, method, df, inv, demean)
   1427     R = toeplitz(r[:-1])
   1428
```

```
-> 1429     rho = np.linalg.solve(R, r[1:])
   1430     sigmasq = r[0] - (r[1:]*rho).sum()
   1431     if inv:

<__array_function__ internals> in solve(*args, **kwargs)

~\.conda\envs\tesis\lib\site-packages\numpy\linalg\linalg.py in solve(a, b)
    392     signature = 'DD->D' if isComplexType(t) else 'dd->d'
    393     extobj = get_linalg_error_extobj(_raise_linalgerror_singular)
--> 394     r = gufunc(a, b, signature=signature, extobj=extobj)
    395
    396     return wrap(r.astype(result_t, copy=False))

~\.conda\envs\tesis\lib\site-packages\numpy\linalg\linalg.py in␣
 ↪_raise_linalgerror_singular(err, flag)
     86
     87 def _raise_linalgerror_singular(err, flag):
---> 88     raise LinAlgError("Singular matrix")
     89
     90 def _raise_linalgerror_nonposdef(err, flag):

LinAlgError: Singular matrix
```

[ ]:
```
fig, axes = plt.subplots(nrows=len(states_list_min), ncols=1, figsize=(10, 10))
crt = 'mean'
for j, i in enumerate(states_list_min):
    pacf_df_dict[i][crt].plot(kind="bar", ax=axes[j])
    axes[j].set_ylabel('Autocorrelación')
    axes[j].set_title(f'Autocorrelación_{i}_{crt}')
```

[ ]:
```
fig, axes = plt.subplots(nrows=len(states_list_min), ncols=1, figsize=(10, 10))
crt = 'abs'
for j, i in enumerate(states_list_min):
    pacf_df_dict[i][crt].plot(kind="bar", ax=axes[j])
    axes[j].set_ylabel('Autocorrelación')
    axes[j].set_title(f'Autocorrelación_{i}_{crt}')
```