Diseño del sistema de potencia

October 2, 2022

1 Diseño del sistema de elevación

1.0.1 1. Cálculo del motor a partir de los momentos máximos que experimenta por la carga de viento

1.0.2 2. Cálculo del eje del eje principal del sistema

1.0.3 3. Cálculo de los elementos de sujeción

1.0.4 4. Cálculo de los rodamientos

Para el desarrollo de los anteriores parámetros se basan en las cargas que calculan en base a los datos experimentales obtenidos en tunel de viento, los cuales se puede obtener a partir del script de Radiotelescope Modelling, el cual se puede obtener en la dirección:

%run C:_Stack_RadiotelescopeModeling_Load_Modelling.ipynb

[1]:
```
import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits import mplot3d #Esto es para realizar gráficas en 3D
import pandas as pd
import numpy as np
import scipy as sci
import sympy as symp
from scipy.interpolate import interp1d,splrep,splev,interpn #Esto es para poder␣
↪realizar la interpolación
#Para realizar la cinemática de multicuerpos
from pytransform3d import rotations as pr_rot
from pytransform3d import plot_utils as pr_plot
import pytransform3d.transformations as pr_trans
import math as mt
```

Definición de nuevas funciones para gráficar A continuación se definen nuevas funciones para gráficar fuerzas, necesarias para represetnar el estado de cargas en el eje de elevación

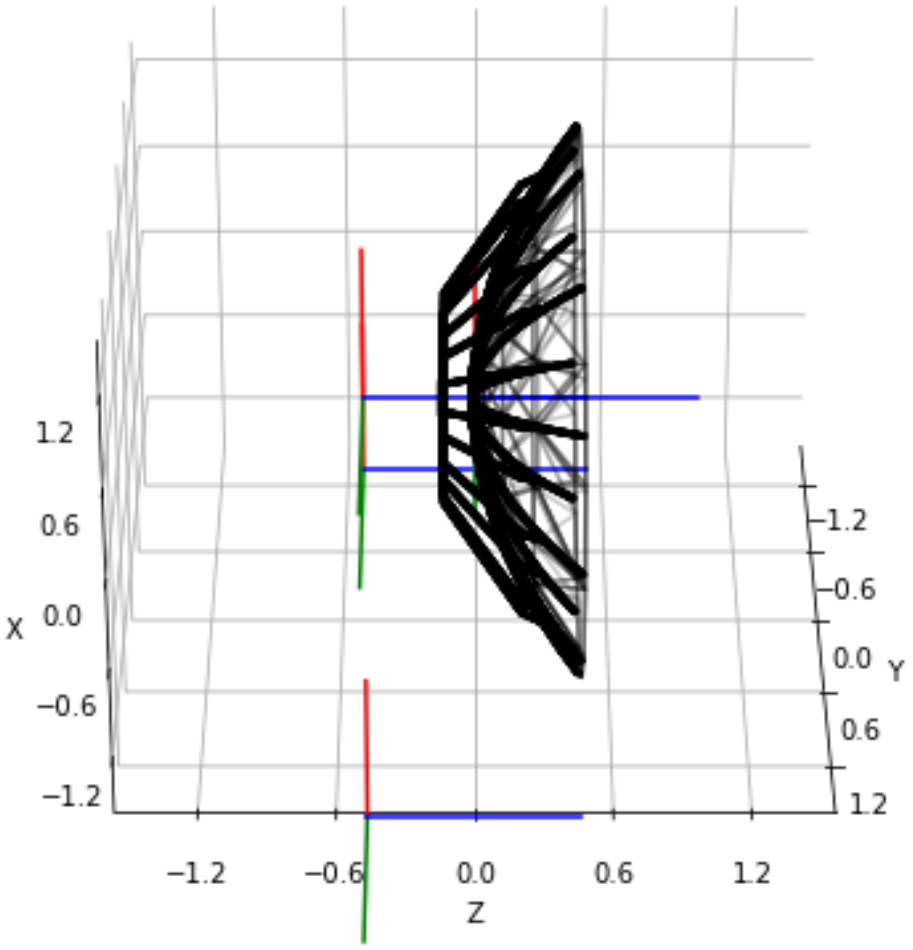
[2]:
```
def plot_forces_1(alpha,beta,Forces,colormap,Load):
    cmap=plt.get_cmap(colormap)
    pallete = []
    for n in np.linspace(0,1,len(alpha)):
```

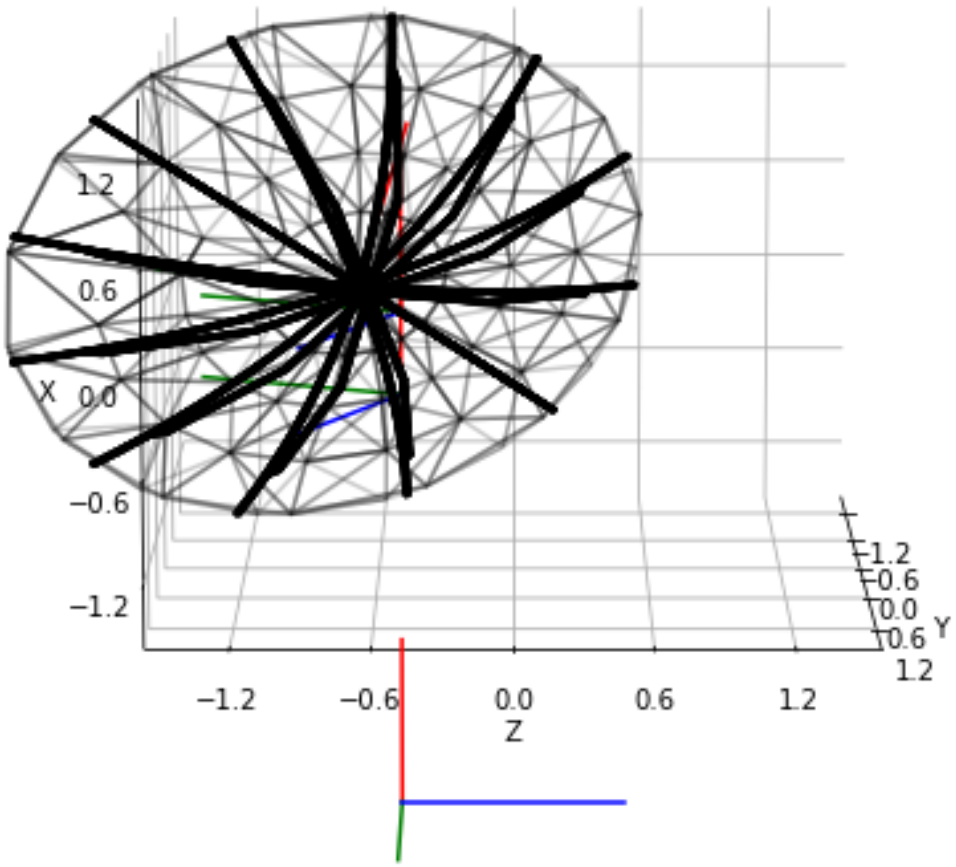
```
        pallete.append(cmap(n))

#Grafica para valores de azimuth 5 a diferentes valores de elevación en el␣
 ↪sistema de coordenadas S0
    fig = plt.figure(figsize =(12,6),tight_layout = True)
    Fx_0 = plt.subplot()

    ilist = range(len(alpha))
    if Load == 0:
        for i in ilist:
            Fx_0.plot(90 - beta,Forces[:,i,0],label = "azimuth: " +␣
 ↪str(alpha[i])+ "°", color = pallete[i])
            Fx_0.set_ylabel("Fuerzas en x [N]")
            Fx_0.set_xlabel("Angulo de elevación")
            Fx_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')
    if Load == 1:
        for i in ilist:
            Fx_0.plot(90 - beta,Forces[:,i,1],label = "azimuth: " +␣
 ↪str(alpha[i])+ "°", color = pallete[i])
            Fx_0.set_ylabel("Fuerzas en y [N]")
            Fx_0.set_xlabel("Angulo de elevación")
            Fx_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')
    if Load == 2:
        for i in ilist:
            Fx_0.plot(90 - beta,Forces[:,i,2],label = "azimuth: " +␣
 ↪str(alpha[i])+ "°", color = pallete[i])
            Fx_0.set_ylabel("Fuerzas en z [N]")
            Fx_0.set_xlabel("Angulo de elevación")
            Fx_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')
```

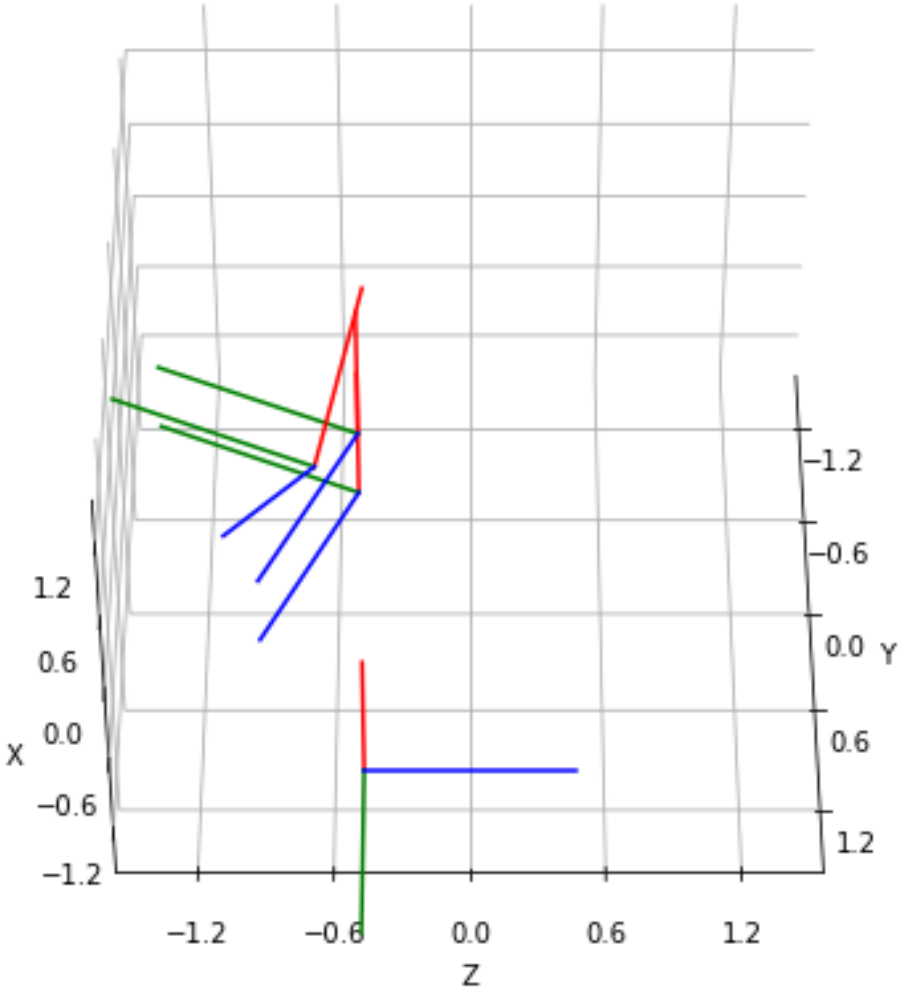
[3]:
```
def plot_forces2(alpha,beta,Forces,Moments,colormap):
    cmap=plt.get_cmap(colormap)
    pallete = []
    for n in np.linspace(0,1,len(alpha)):
        pallete.append(cmap(n))

#Grafica para valores de azimuth 5 a diferentes valores de elevación en el␣
 ↪sistema de coordenadas S0
    fig = plt.figure(figsize =(15,13),tight_layout = True)
    gs = fig.add_gridspec(3,2)

    Fx_0= fig.add_subplot(gs[0,0])
    Fy_0 = fig.add_subplot(gs[1,0])
    Fz_0 = fig.add_subplot(gs[2,0])
    Mx_0 = fig.add_subplot(gs[0,1])
    My_0 = fig.add_subplot(gs[1,1])
    Mz_0 = fig.add_subplot(gs[2,1])
```

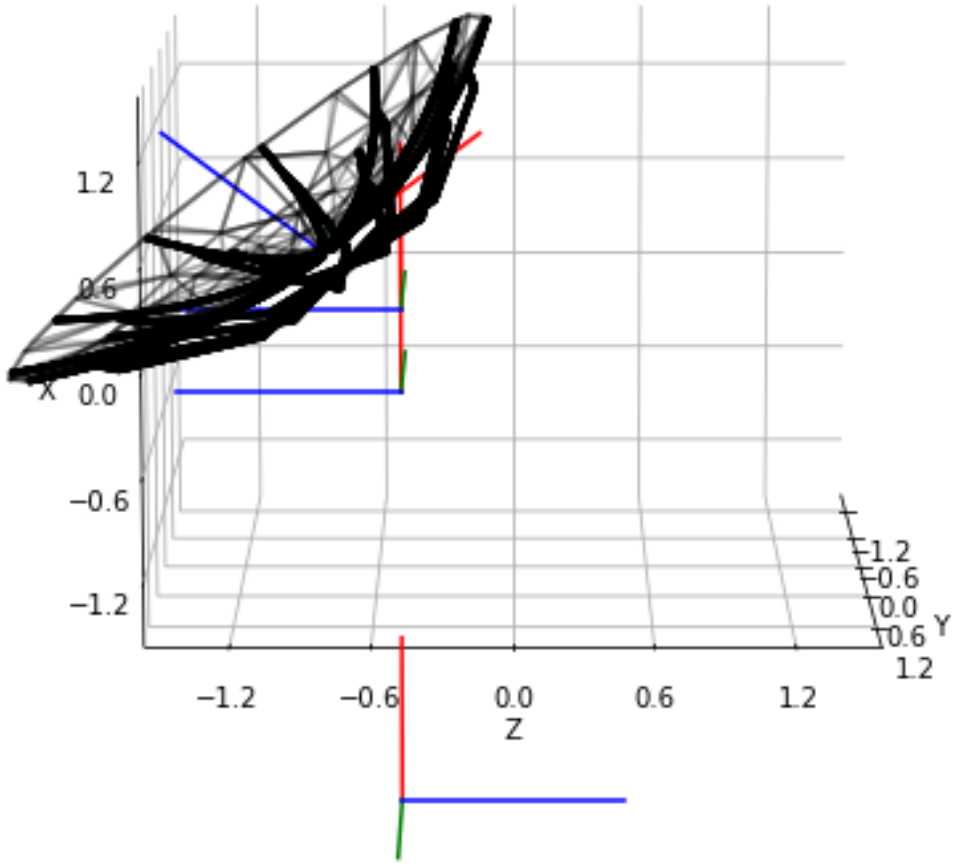
```
    ilist = range(len(alpha))

    for i in ilist:
        Fx_0.plot(90 - beta,Forces[:,i,0],label = "azimuth: " + str(alpha[i])+␣
↪"°", color = pallete[i])
        Fx_0.set_ylabel("Fuerzas en x [N]")
        Fx_0.set_xlabel("Angulo de elevación")
        Fx_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')

    for i in ilist:
        Fy_0.plot(90 - beta,Forces[:,i,1],label = "azimuth: " + str(alpha[i])+␣
↪"°",color = pallete[i])
        Fy_0.set_ylabel("Fuerzas en y [N]")
        Fy_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')

    for i in ilist:
        Fz_0.plot(90 - beta,Forces[:,i,2],label = "azimuth: " + str(alpha[i])+␣
↪"°",color = pallete[i])
        Fz_0.set_ylabel("Fuerzas en z [N]")
        Fz_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')

    for i in ilist:
        Mx_0.plot(90 - beta,Moments[:,i,0],label = "azimuth: " + str(alpha[i])+␣
↪"°",color = pallete[i])
        Mx_0.set_ylabel("Fuerzas en x [Nm]")
        Mx_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')

    for i in ilist:
        My_0.plot(90 - beta,Moments[:,i,1],label = "azimuth: " + str(alpha[i])+␣
↪"°",color = pallete[i])
        My_0.set_ylabel("Fuerzas en y [Nm]")
        My_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')

    for i in ilist:
        Mz_0.plot(90 - beta,Moments[:,i,2],label = "azimuth: " + str(alpha[i])+␣
↪"°",color = pallete[i])
        Mz_0.set_ylabel("Fuerzas en z [Nm]")
        Mz_0.legend(bbox_to_anchor=(1.05, 1.0), loc = 2, fontsize='small')
```

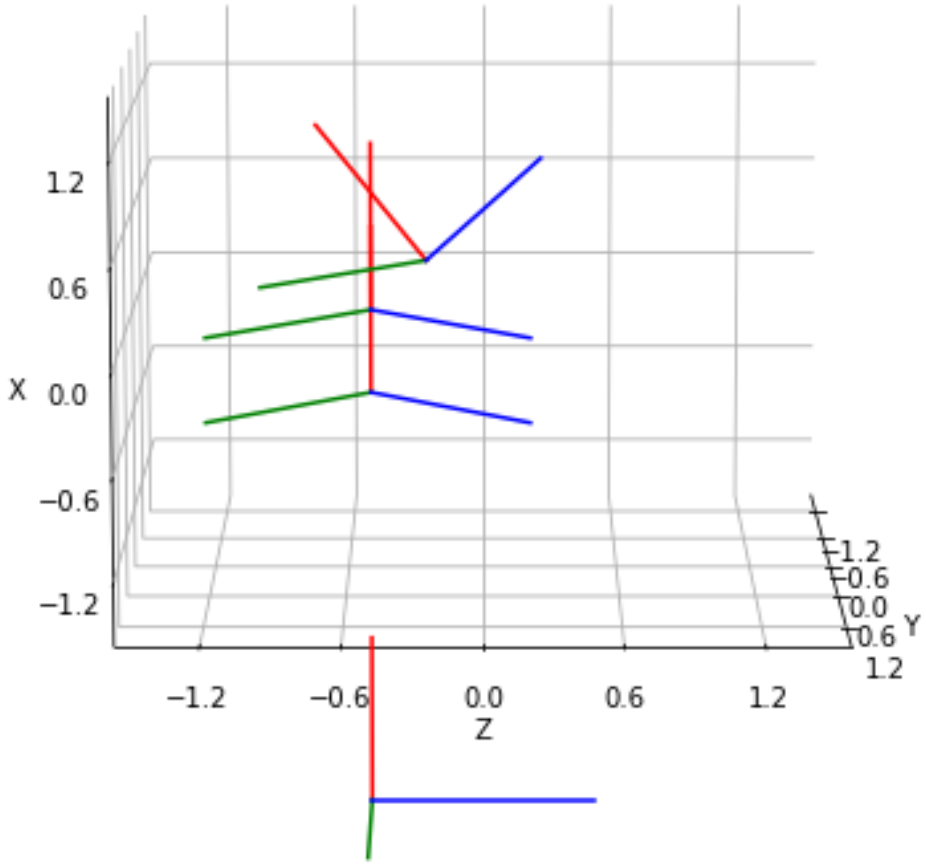
[4]:
```
%run C:
↪\Users\David\Documents\Hay_Stack_Radiotelescope_Notebooks\Antenna_Modeling\Radiotelescope_L
↪ipynb
```


1.2
0.6
0.0
−0.6
−1.2
X
−1.2
−0.6
0.0
0.6
1.2
Z
−1.2
−0.6
0.0
0.6
1.2
Y

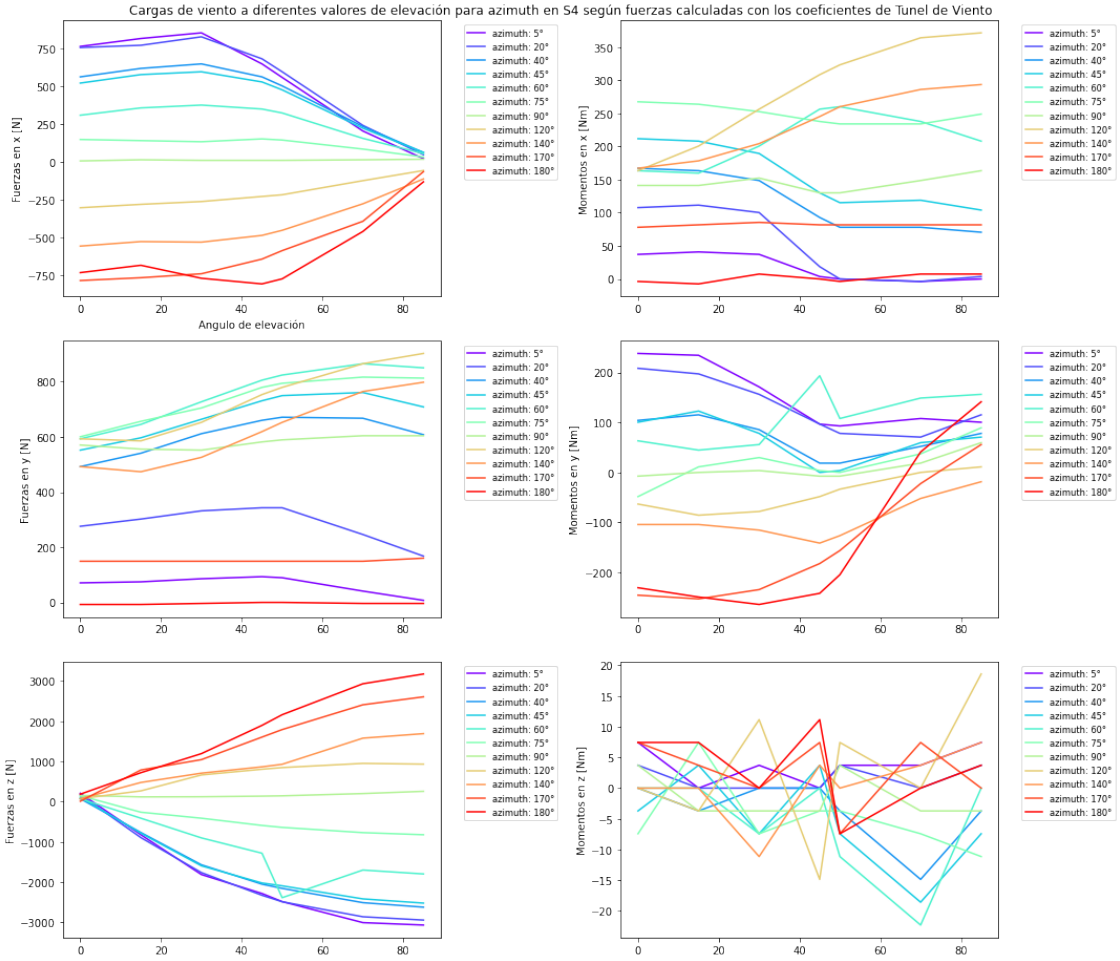

1.2
0.6
X 0.0
−0.6
−1.2
−1.2
−0.6
0.0
0.6
1.2
Z
1.2
0.6
0.0
0.6
1.2
Y

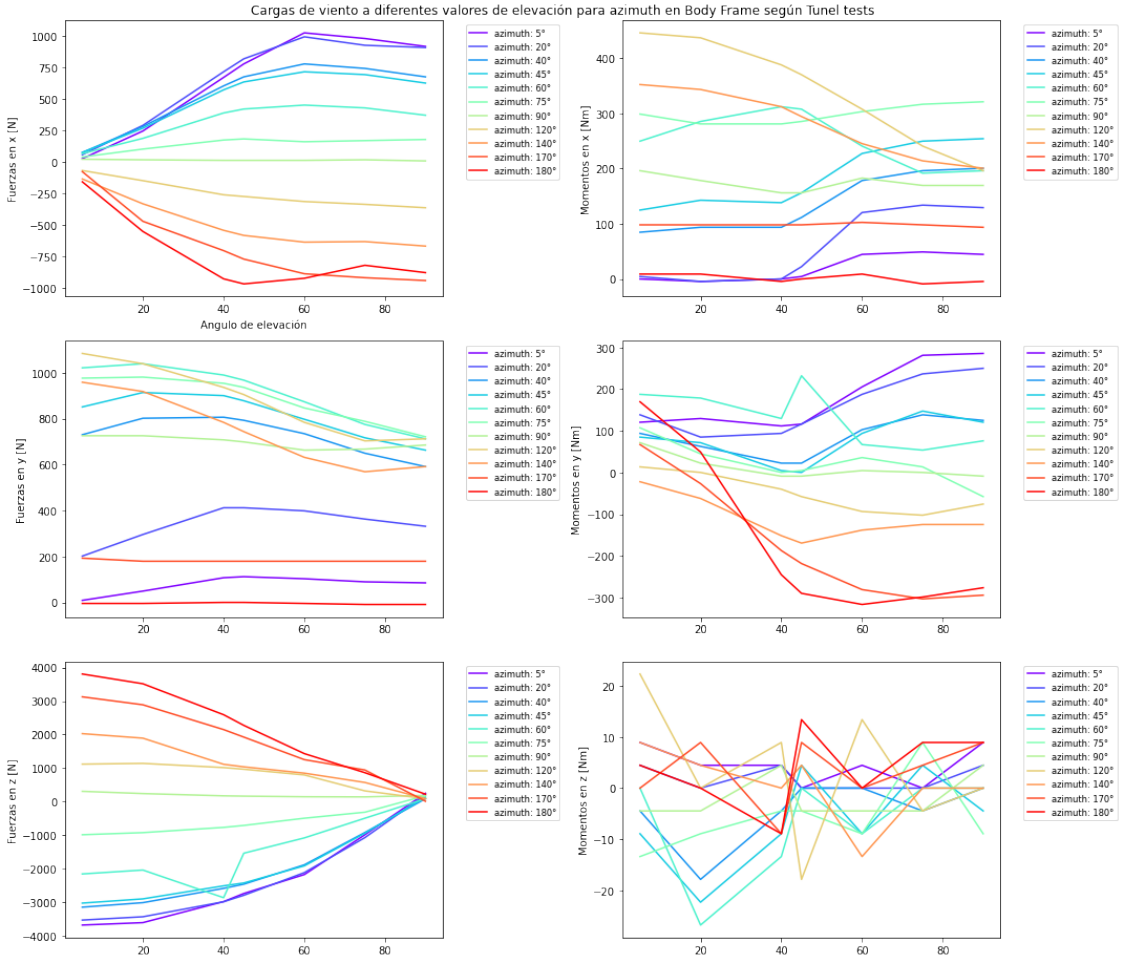

1.2
0.6
0.0
X
−0.6
−1.2
−1.2
−0.6
0.0
0.6
1.2
Z
−1.2
−0.6
0.0
0.6
1.2
Y

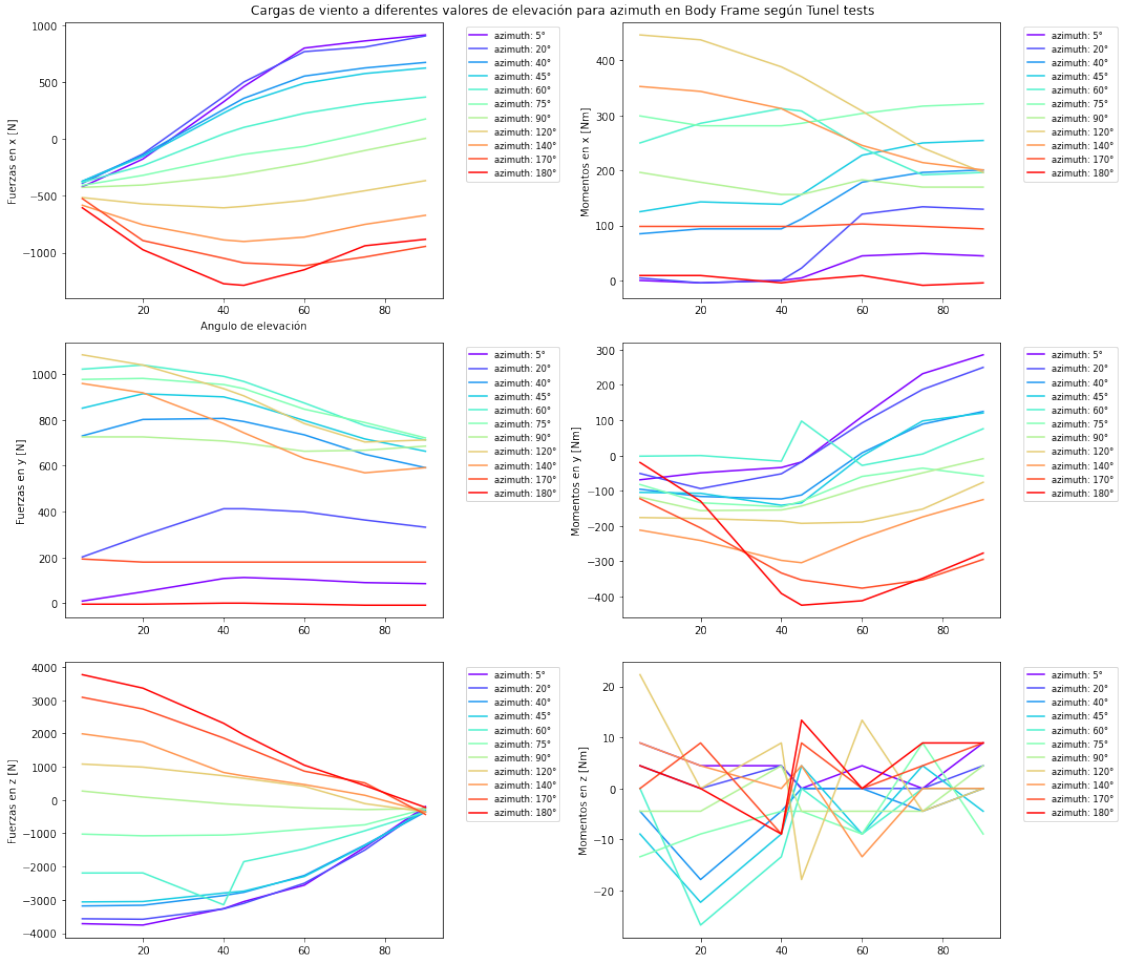

1.2
0.6
X 0.0
−0.6
−1.2
−1.2
−0.6
0.0
0.6
1.2
Z
1.2
0.6
0.0
0.6
1.2
Y




```
-----------------------------------------DATOS CARGADOS
CORRECTAMENTE-----------------------------------
 Angulos de azimuth alpha:  5 - 180 ,y contiene:  11
 Angulos de cenitales beta 5 - 90 , y contiene:  7
 ----------------------------Las cargas en {C} se pueden obtener con las
variables:----------------------
 Forces_S4
 Moments_S4
 Wrench: Es el arreglo que contiene para cada configuración el vector Wrench
[Momentos + Fuerzas]
 New_Wrench: Es un dataframe que contiene el vector Wrench para cada
configuración
             Se puede acceder mediante New_Wrench.loc[beta,alpha]
```

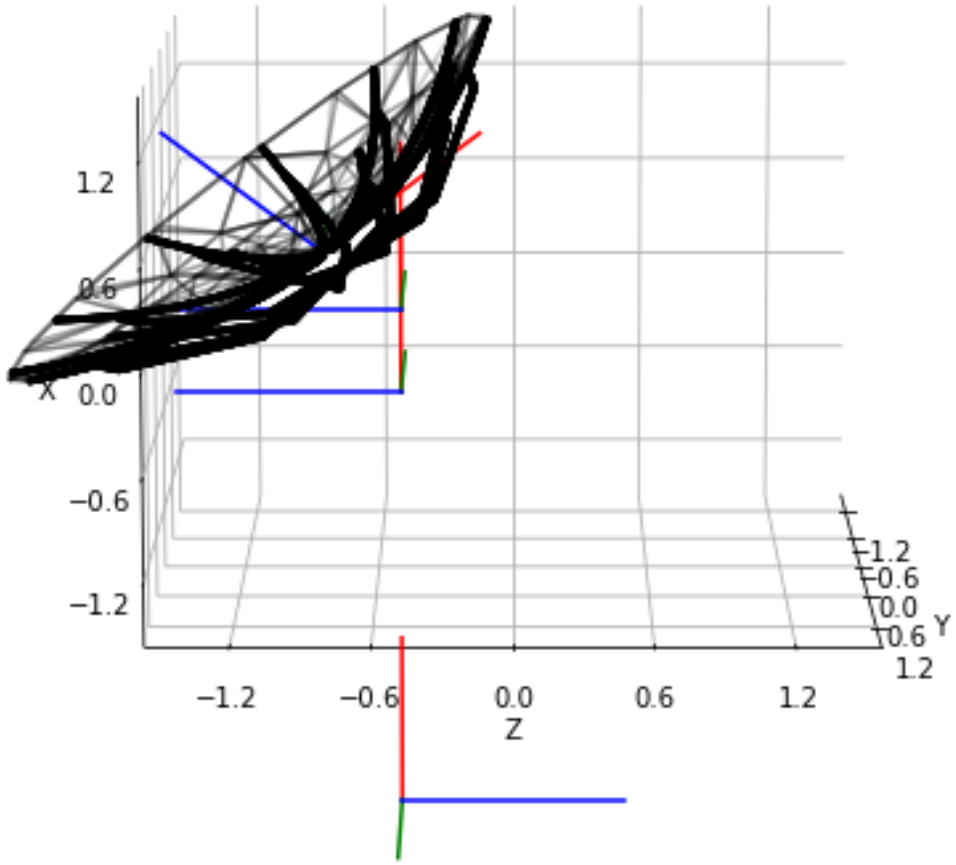

Cargas de viento a diferentes valores de elevación para azimuth en S4 según fuerzas calculadas con los coeficientes de Tunel de Viento
Fuerzas en x [N]
Momentos en x [Nm]
Fuerzas en y [N]
Momentos en y [Nm]
Fuerzas en z [N]
Momentos en z [Nm]
Angulo de elevación
azimuth: 5°
azimuth: 20°
azimuth: 40°
azimuth: 45°
azimuth: 60°
azimuth: 75°
azimuth: 90°
azimuth: 120°
azimuth: 140°
azimuth: 170°
azimuth: 180°


Cargas de viento a diferentes valores de elevación para azimuth en Body Frame según Tunel tests
Fuerzas en x [N]
Momentos en x [Nm]
Fuerzas en y [N]
Momentos en y [Nm]
Fuerzas en z [N]
Momentos en z [Nm]
Angulo de elevación
azimuth: 5°
azimuth: 20°
azimuth: 40°
azimuth: 45°
azimuth: 60°
azimuth: 75°
azimuth: 90°
azimuth: 120°
azimuth: 140°
azimuth: 170°
azimuth: 180°



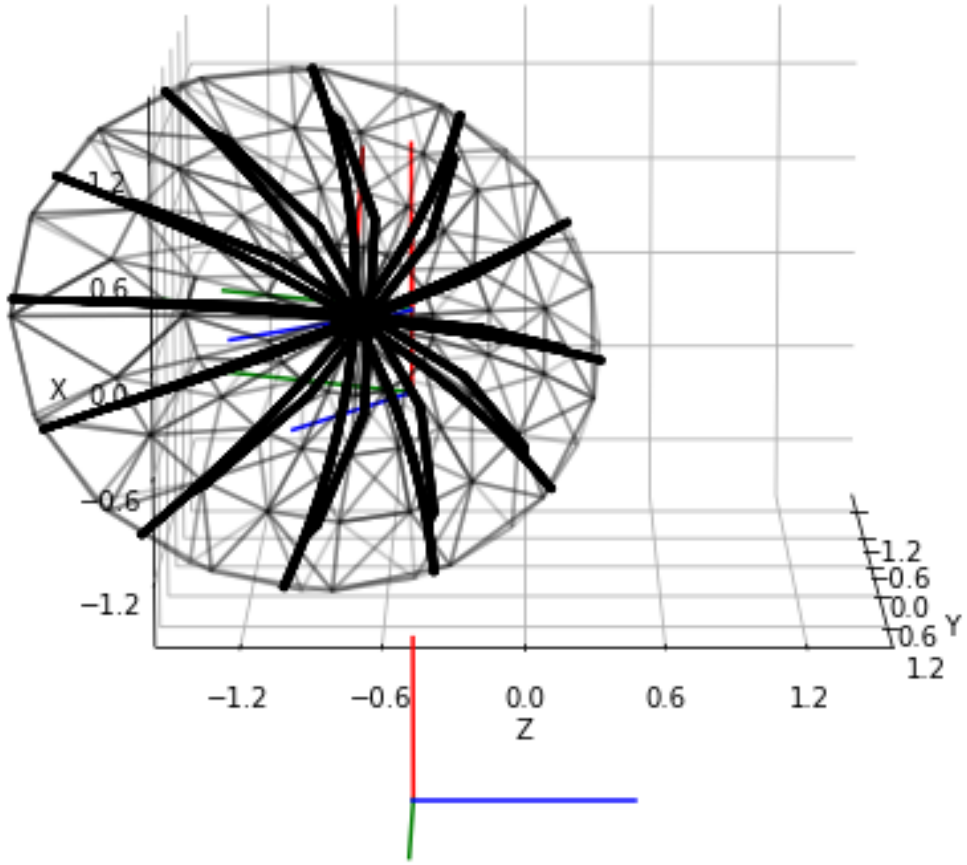
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN X----------

Azimuth: [180]
Elevación: [45]
Fuerza en x: -1286.841341055839
Fuerza en y: 0.0
Fuerza en z: 1959.2607118455176
Momentos en x: 0.0
Momentos en y: -424.2186341483003
Momentos en z: 13.381033053372672



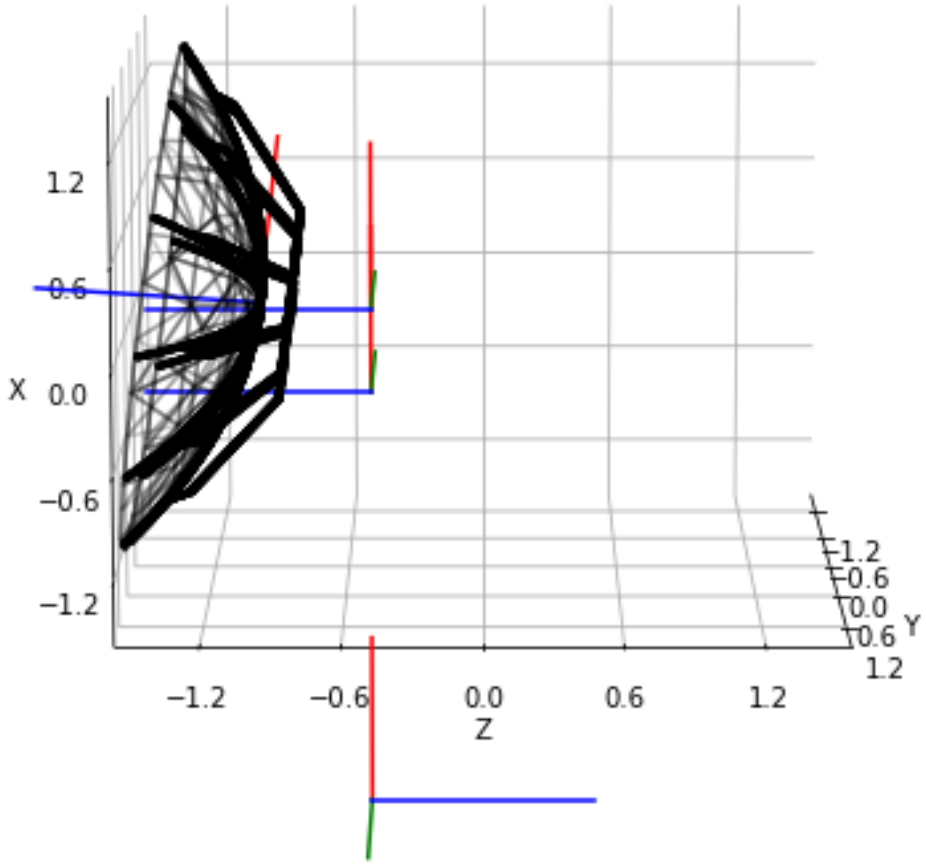
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Y ----------

Azimuth: [120]
Elevación: [5]
Fuerza en x: -515.7715965211694
Fuerza en y: 1083.526478347763
Fuerza en z: 1080.488890694038
Momentos en x: 446.0344351124224
Momentos en y: -175.8198303852308
Momentos en z: 22.30172175562112



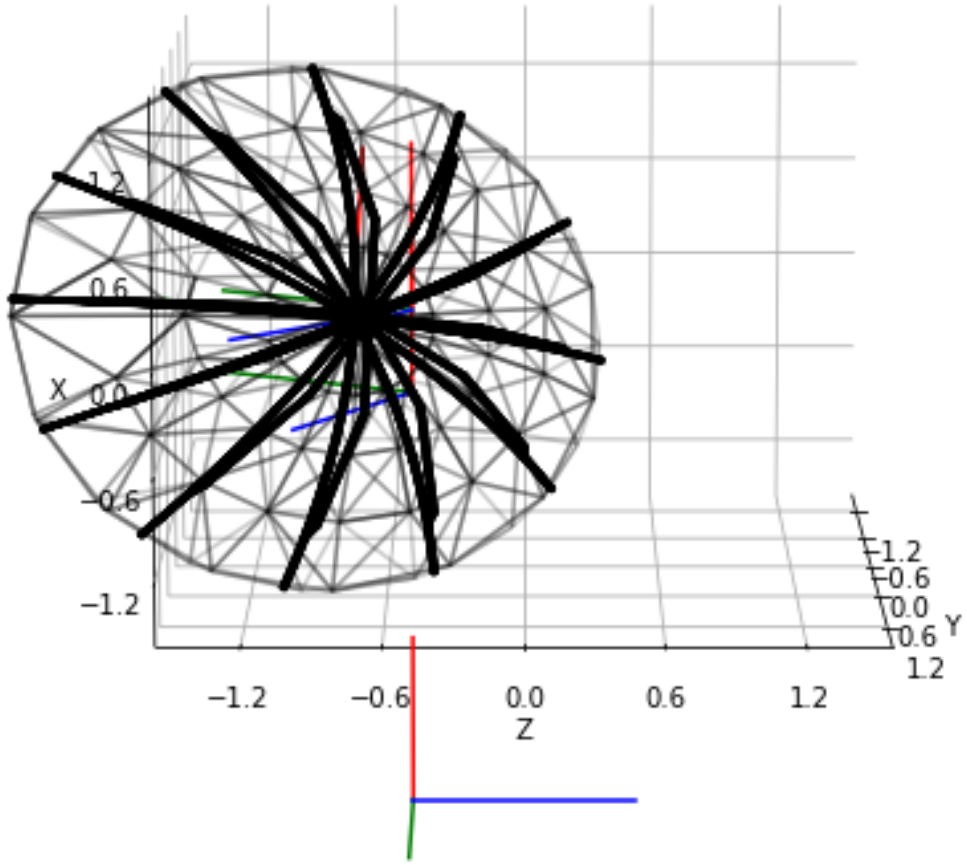
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Z ----------

Azimuth: [180]
Elevación: [5]
Fuerza en x: -605.3192393598274
Fuerza en y: -4.477382141932906
Fuerza en z: 3766.9181758537807
Momentos en x: 8.920688702248448
Momentos en y: -19.707778095882958
Momentos en z: 4.460344351124224



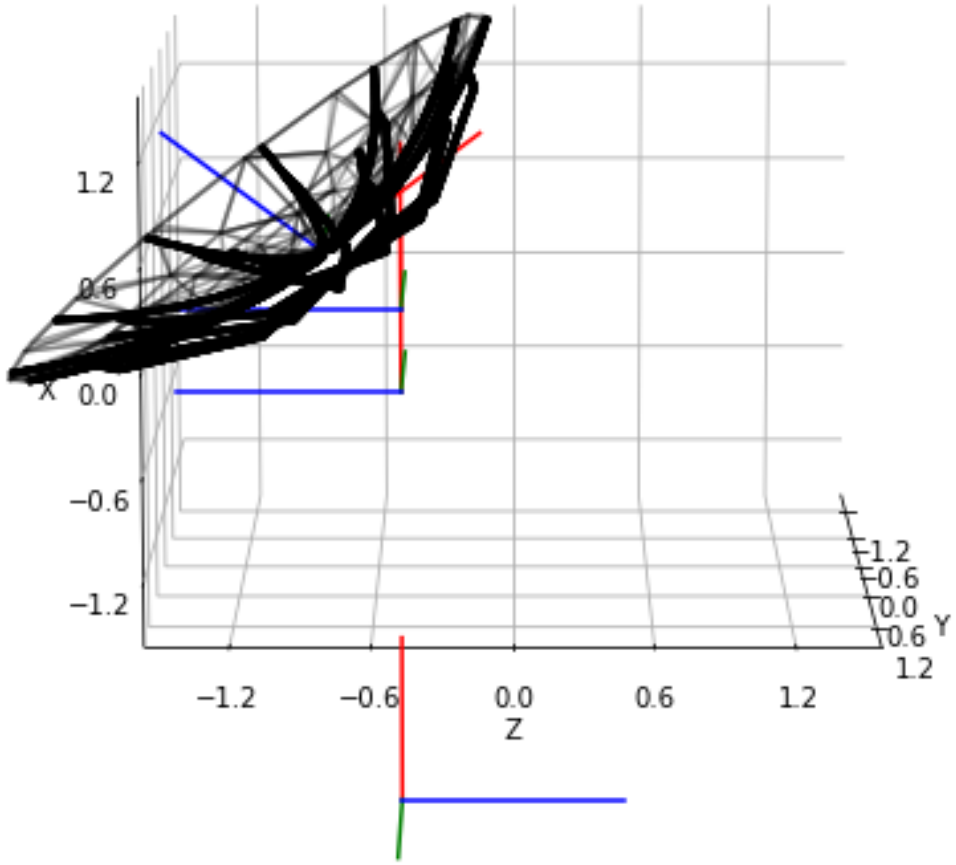
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN X ----------

Azimuth: [120]
Elevación: [5]
Fuerza en x: -515.7715965211694
Fuerza en y: 1083.526478347763
Fuerza en z: 1080.488890694038
Momentos en x: 446.0344351124224
Momentos en y: -175.8198303852308
Momentos en z: 22.30172175562112



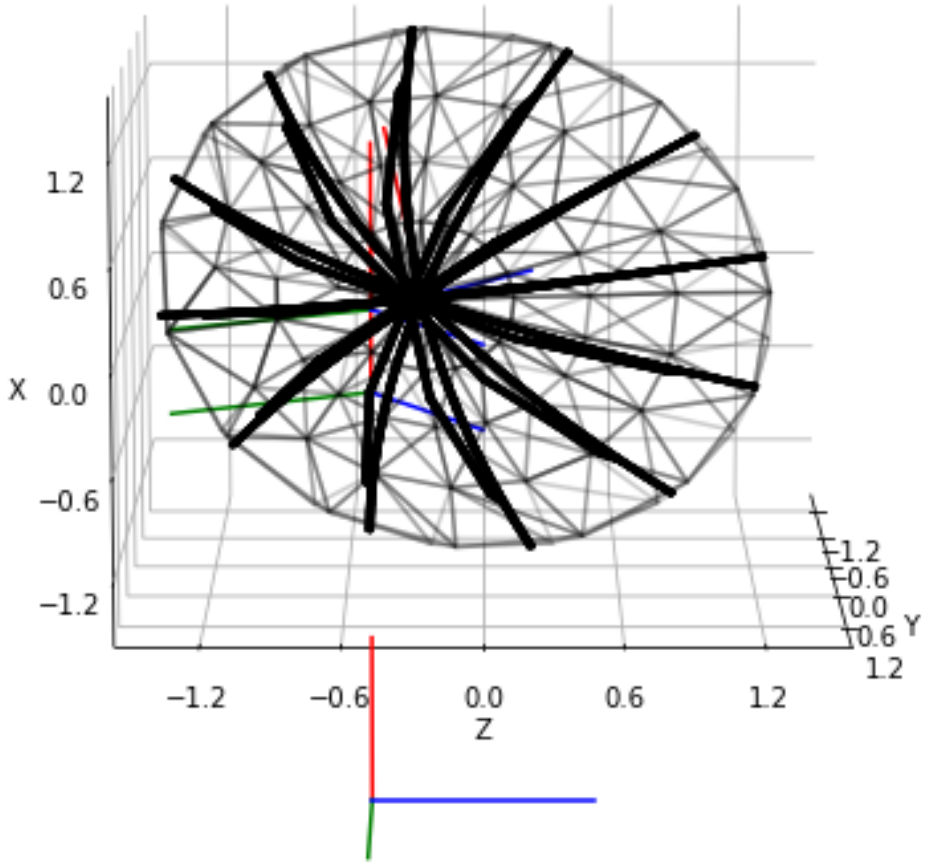
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Y ----------

Azimuth: [180]
Elevación: [45]
Fuerza en x: -1286.841341055839
Fuerza en y: 0.0
Fuerza en z: 1959.2607118455176
Momentos en x: 0.0
Momentos en y: -424.2186341483003
Momentos en z: 13.381033053372672

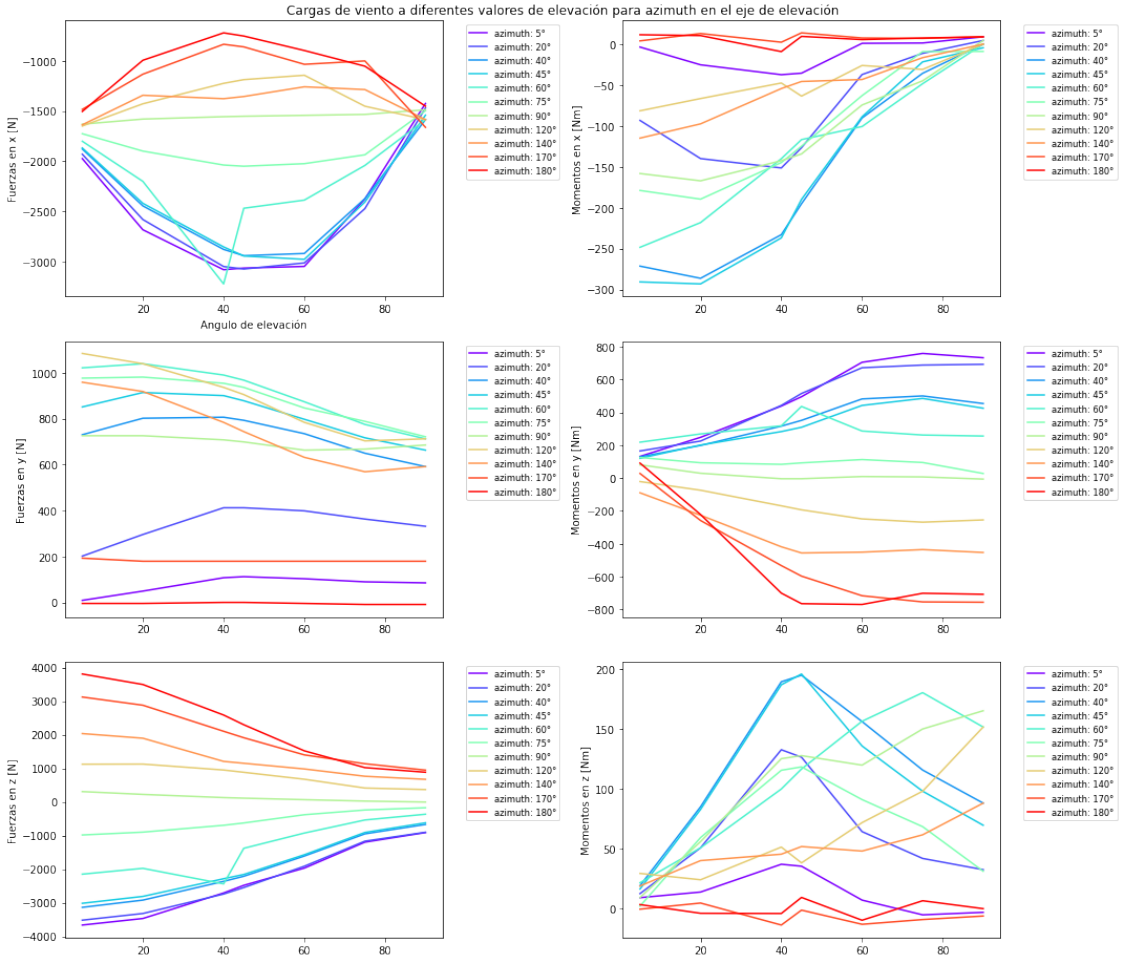


----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Z ----------

Azimuth: [60]
Elevación: [20]
Fuerza en x: -235.37049330573242
Fuerza en y: 1038.7526569284341
Fuerza en z: -2189.6917150783547
Momentos en x: 285.46203847195034
Momentos en y: -0.05601259427317018
Momentos en z: -26.762066106745344

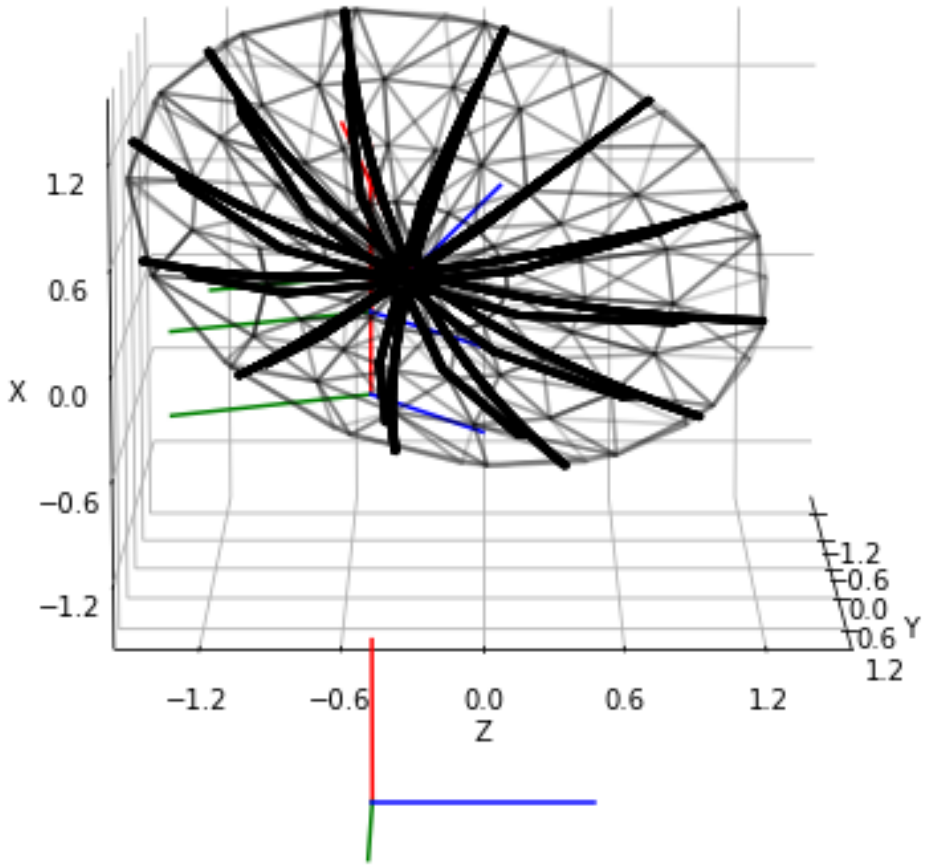



```
[[[   22.38691071     8.95476428 -3675.93073853]]]
[[[ -421.74657154     8.95476428 -3714.78738332]]]
```



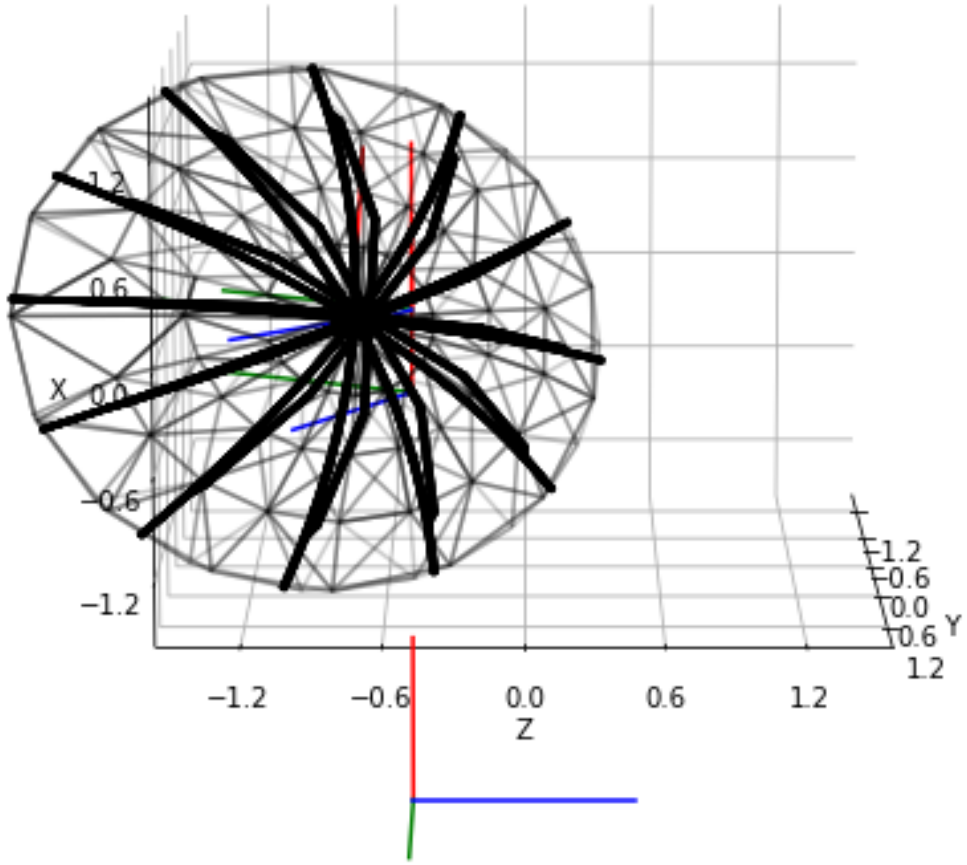
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN X ----------

Azimuth: [60]
Elevación: [40]
Fuerza en x: -3219.9765446006186
Fuerza en y: 989.5014533671721
Fuerza en z: -2439.1977961623734
Momentos en x: -140.086643848332
Momentos en y: 317.6418147794489
Momentos en z: 100.07895309932951



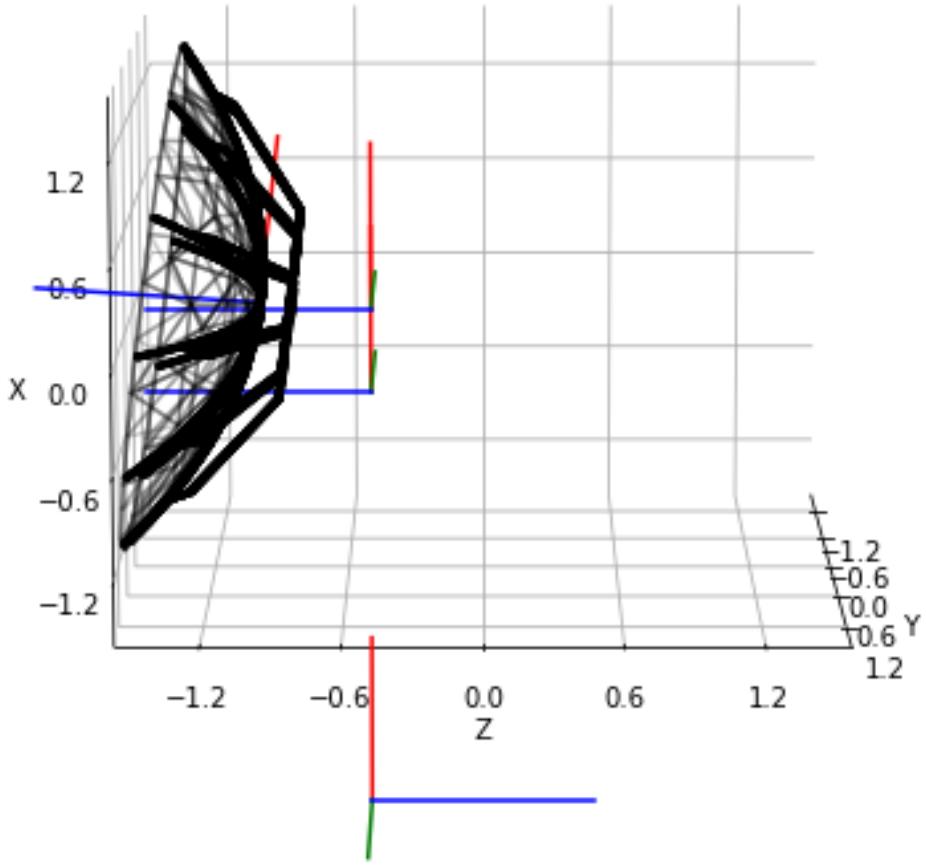
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Y ----------

Azimuth: [120]
Elevación: [5]
Fuerza en x: -1649.708118081672
Fuerza en y: 1083.526478347763
Fuerza en z: 1121.32976083938
Momentos en x: -81.54736727383367
Momentos en y: -21.650004825110383
Momentos en z: 29.521380886509903



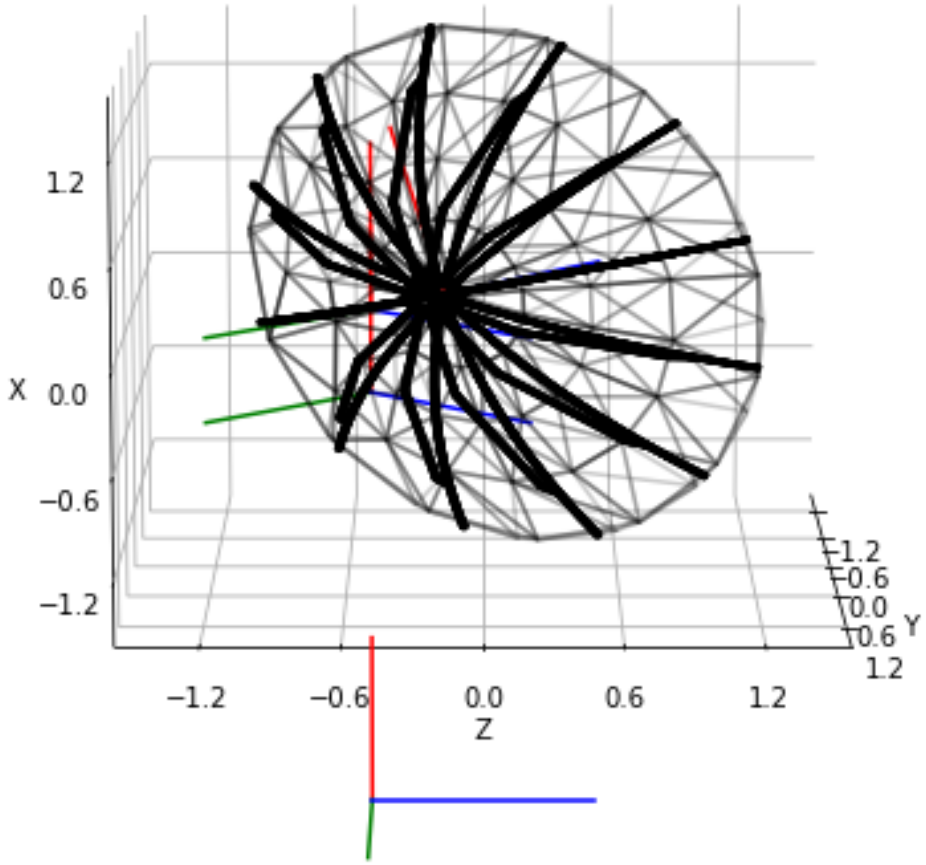
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Z ----------

Azimuth: [180]
Elevación: [5]
Fuerza en x: -1504.777265416999
Fuerza en y: -4.477382141932906
Fuerza en z: 3805.340962836819
Momentos en x: 11.456595801039137
Momentos en y: 90.67325011613367
Momentos en z: 3.47505988674328



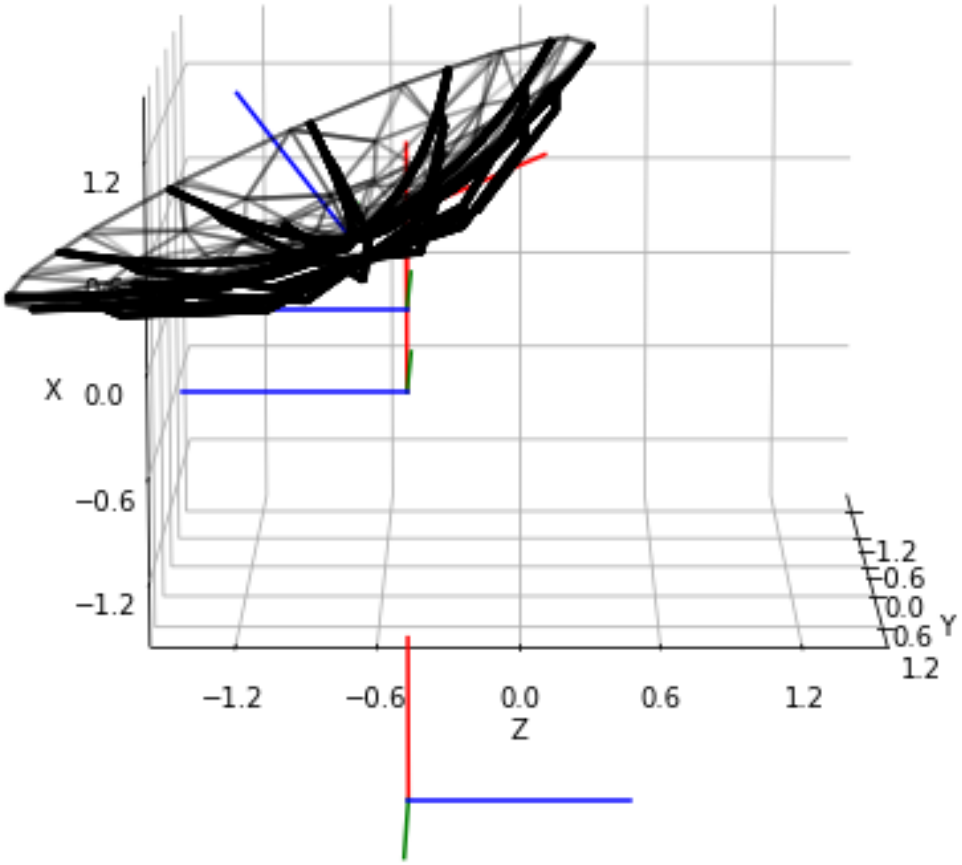
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN X ----------

Azimuth: [45]
Elevación: [20]
Fuerza en x: -2418.452194000706
Fuerza en y: 913.3859569543126
Fuerza en z: -2812.513772643336
Momentos en x: -293.21405189721685
Momentos en y: 200.54246179489382
Momentos en z: 82.988190578898



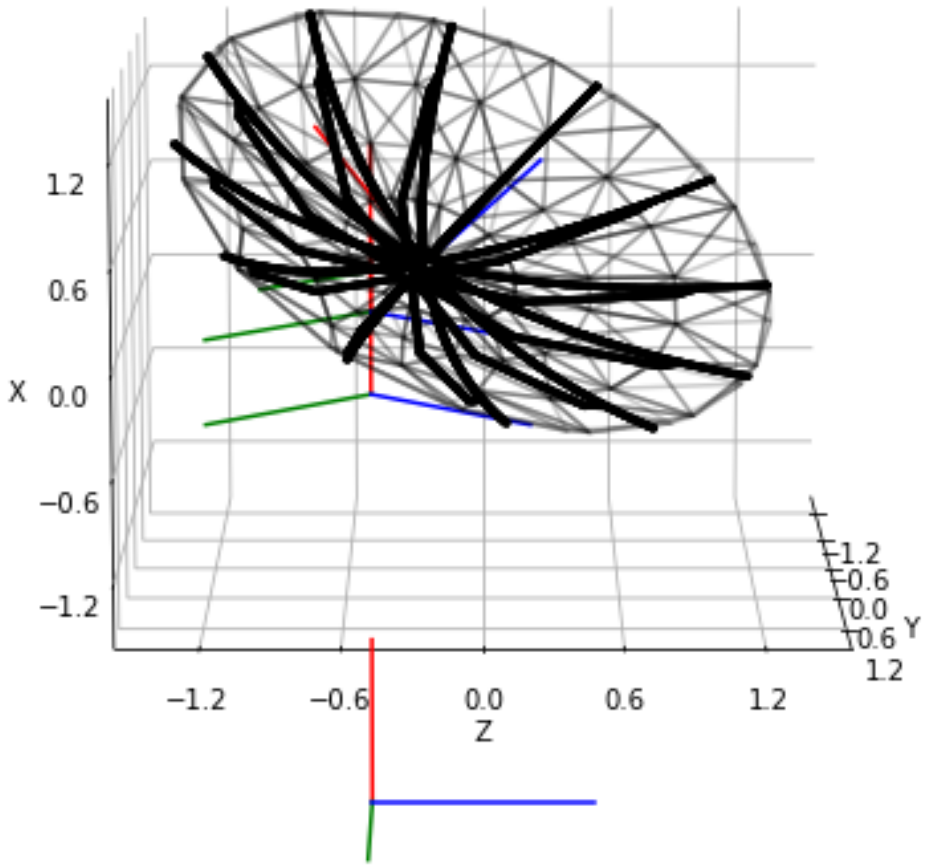
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Y ----------

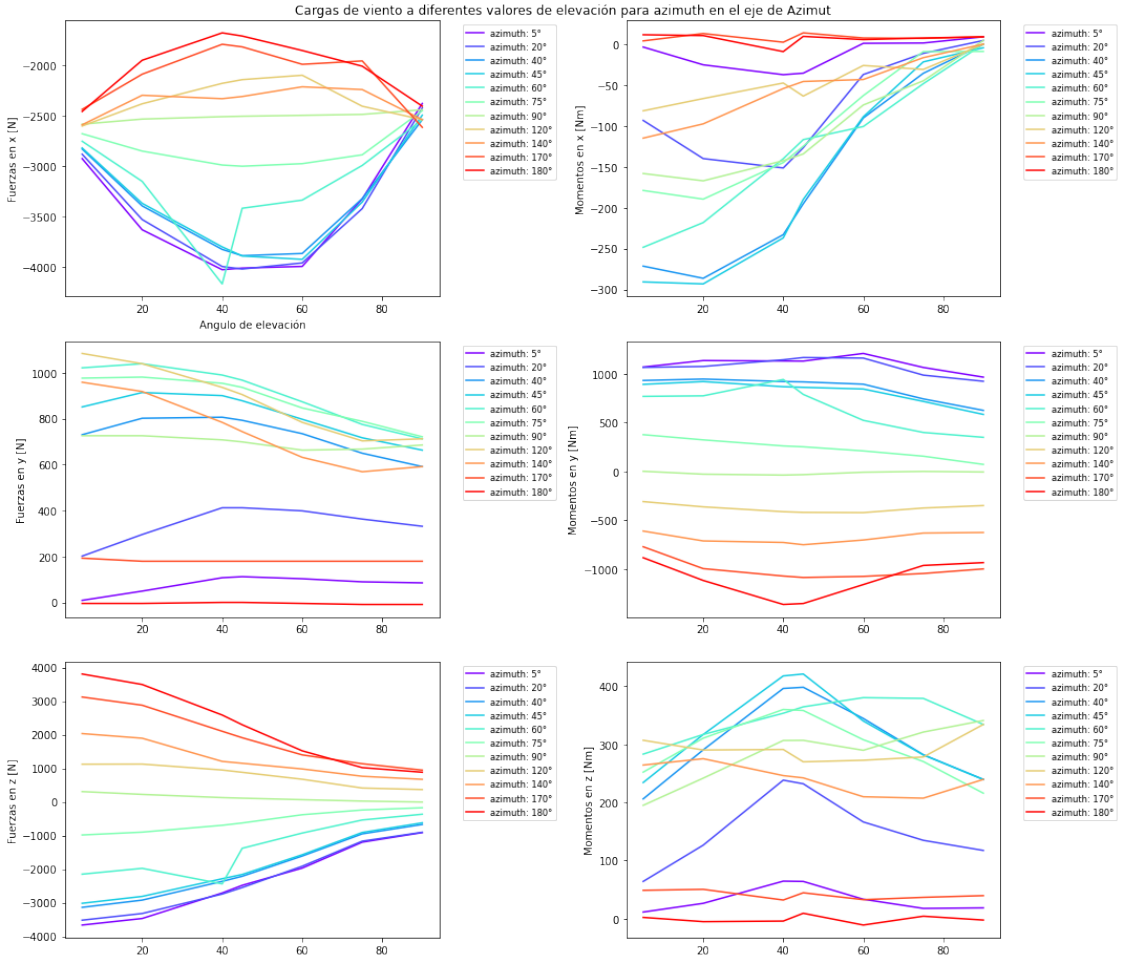
Azimuth: [180]
Elevación: [60]
Fuerza en x: -898.5005149333588
Fuerza en y: -4.477382141932906
Fuerza en z: 1519.0291649237538
Momentos en x: 5.555064284826823
Momentos en y: -769.8985014826943
Momentos en z: -9.62165358063132



----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Z ----------

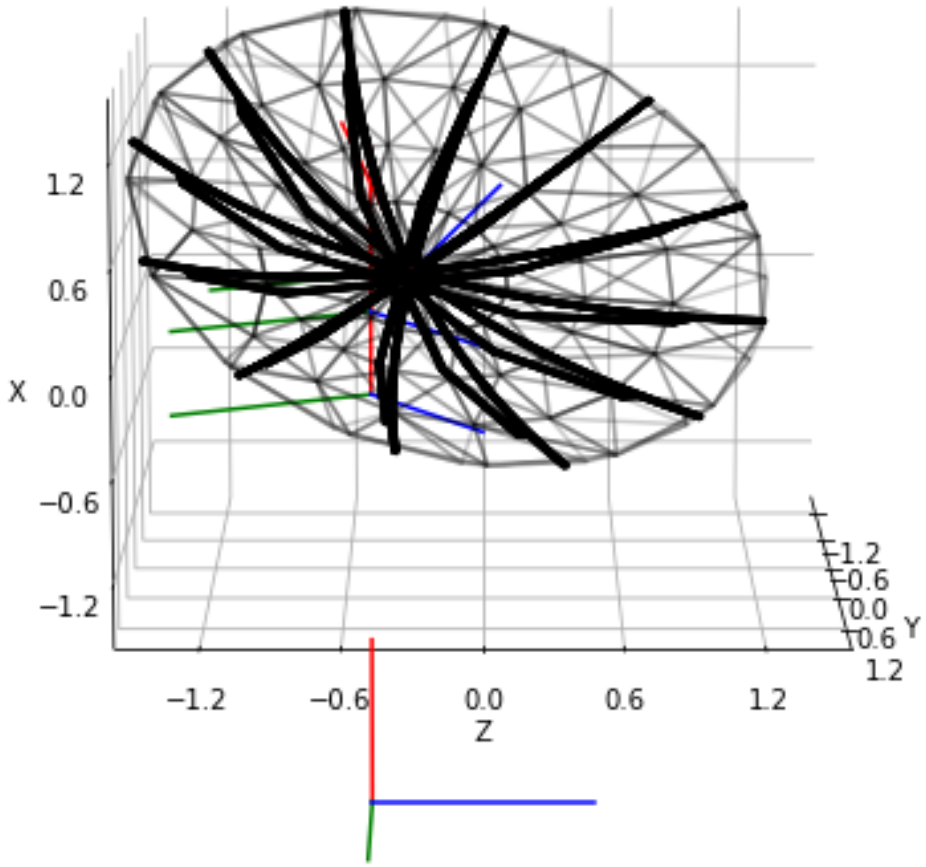
Azimuth: [45]
Elevación: [45]
Fuerza en x: -2942.2949098097233
Fuerza en y: 877.5668998188495
Fuerza en z: -2159.2033212255774
Momentos en x: -189.89905380365383
Momentos en y: 308.7110213041318
Momentos en z: 196.20693327786793


1.2
0.6
X 0.0
−0.6
−1.2
−1.2
−0.6
0.0
0.6
1.2
Z
1.2
0.6
0.0
0.6
1.2
Y



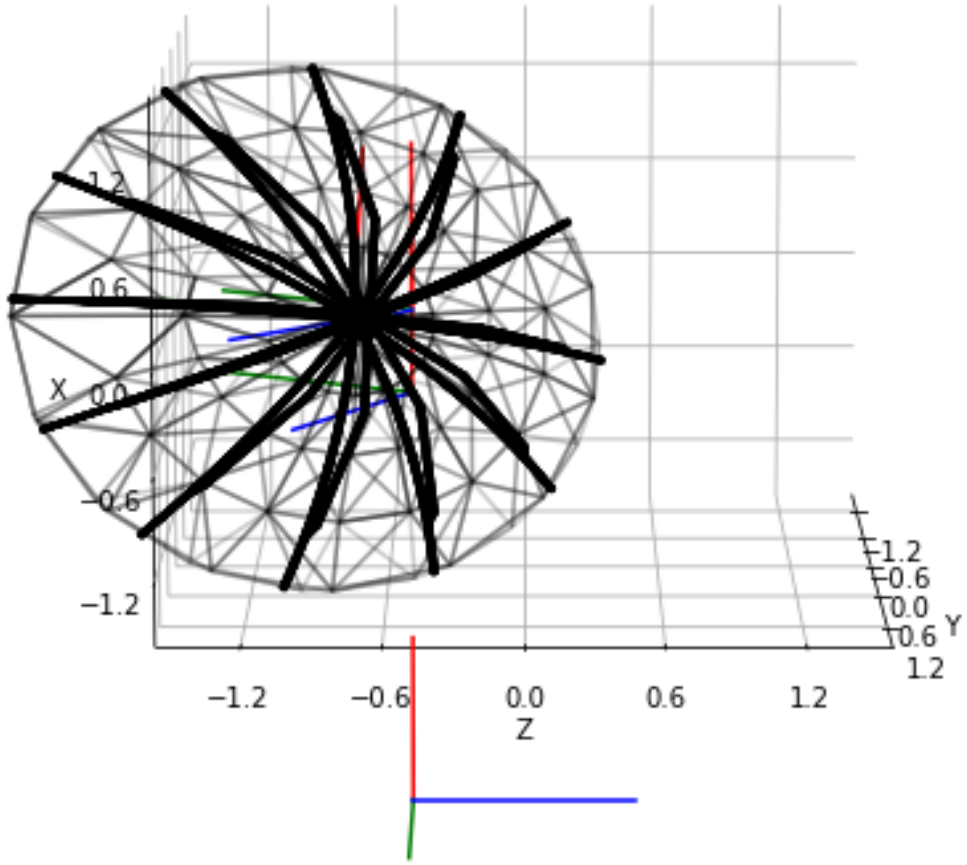
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN X ----------

Azimuth: [60]
Elevación: [40]
Fuerza en x: -4169.976544600619
Fuerza en y: 989.5014533671721
Fuerza en z: -2439.1977961623734
Momentos en x: -140.086643848332
Momentos en y: 942.0764505970166
Momentos en z: 353.39132516132554



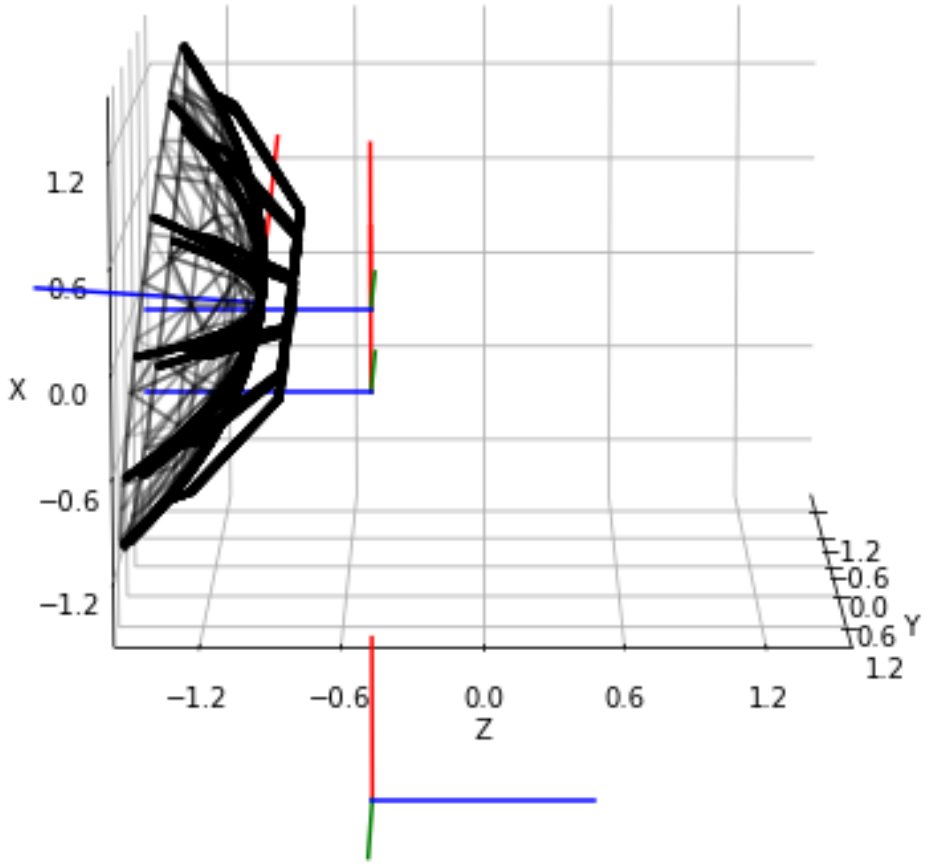
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Y ----------

Azimuth: [120]
Elevación: [5]
Fuerza en x: -2599.708118081672
Fuerza en y: 1083.526478347763
Fuerza en z: 1121.32976083938
Momentos en x: -81.54736727383367
Momentos en y: -308.71042359999166
Momentos en z: 306.90415934353723



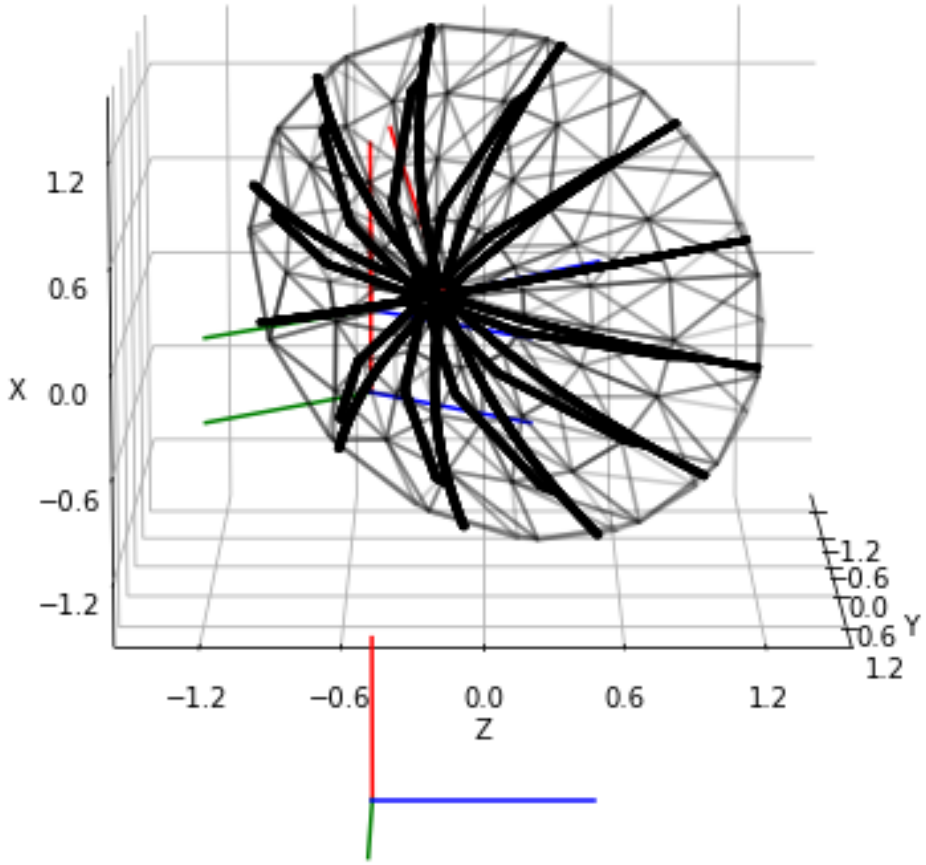
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN Z ----------

Azimuth: [180]
Elevación: [5]
Fuerza en x: -2454.777265416999
Fuerza en y: -4.477382141932906
Fuerza en z: 3805.340962836819
Momentos en x: 11.456595801039137
Momentos en y: -883.494036370092
Momentos en z: 2.328850058408456



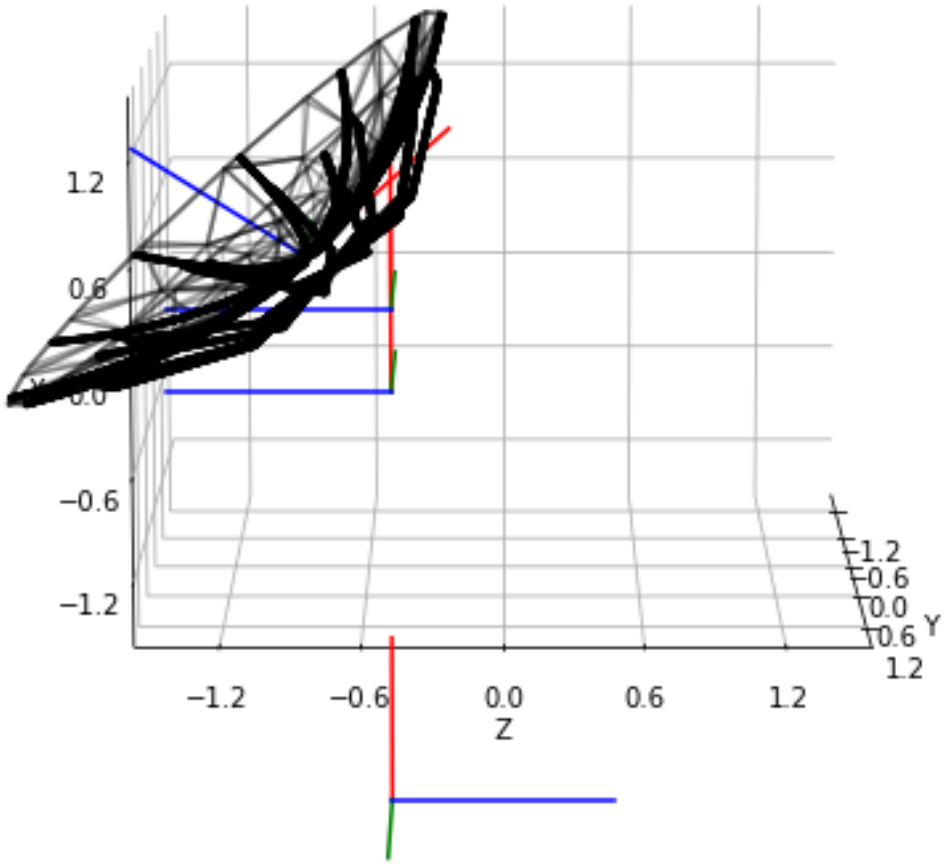
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN X ----------

Azimuth: [45]
Elevación: [20]
Fuerza en x: -3368.452194000706
Fuerza en y: 913.3859569543126
Fuerza en z: -2812.513772643336
Momentos en x: -293.21405189721685
Momentos en y: 920.5459875915878
Momentos en z: 316.81499555920203



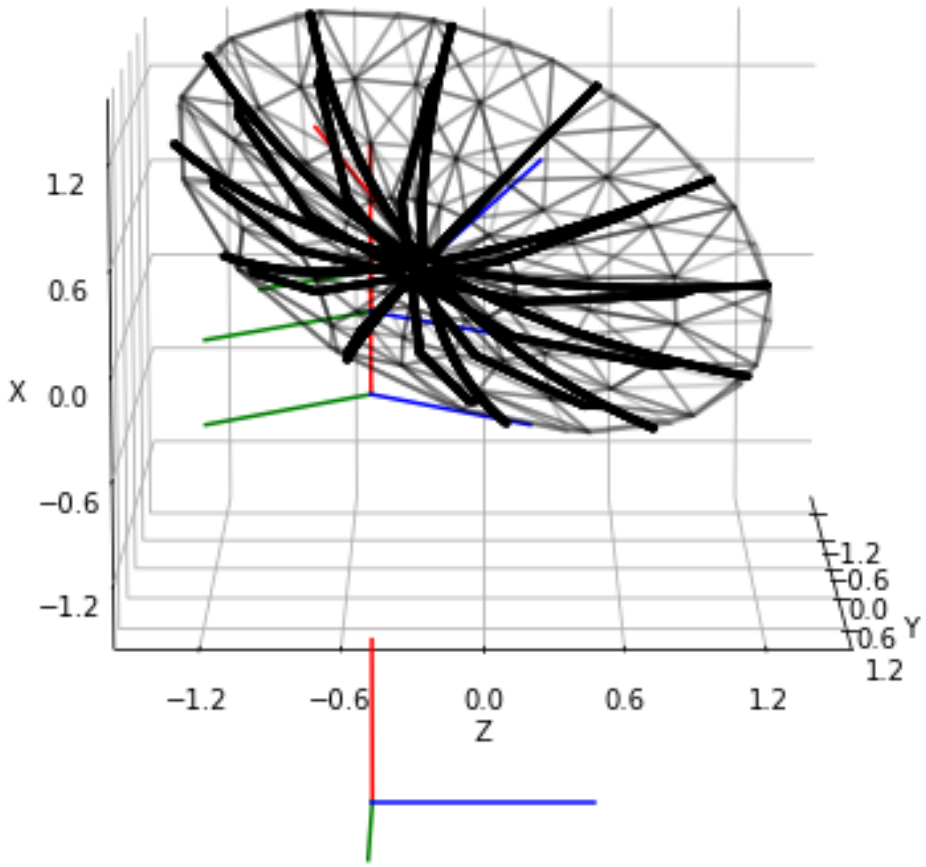
----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Y ----------

Azimuth: [180]
Elevación: [40]
Fuerza en x: -1672.948393844194
Fuerza en y: 0.0
Fuerza en z: 2584.5220768787844
Momentos en x: -9.150930172251833
Momentos en y: -1362.360083413081
Momentos en z: -3.9665899253136727



----------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Z ----------

Azimuth: [45]
Elevación: [45]
Fuerza en x: -3892.2949098097233
Fuerza en y: 877.5668998188495
Fuerza en z: -2159.2033212255774
Momentos en x: -189.89905380365383
Momentos en y: 861.4670715378796
Momentos en z: 420.86405963149343



1.1 Definición dinámica del sistema Acople-Antena

Modelado dinámico EL sistema Acople-Antena se define dinámicamente con su matriz de inercia y peso del sistema.

Matriz de Inercia Ixx = 107.663 Kgm^2 Iyy = 94.231 Kgm^2 Izz = 45,165 Kgm^2

En los ejes definidos en:

Frame C: {O_b, x_c,y_c,z_c}

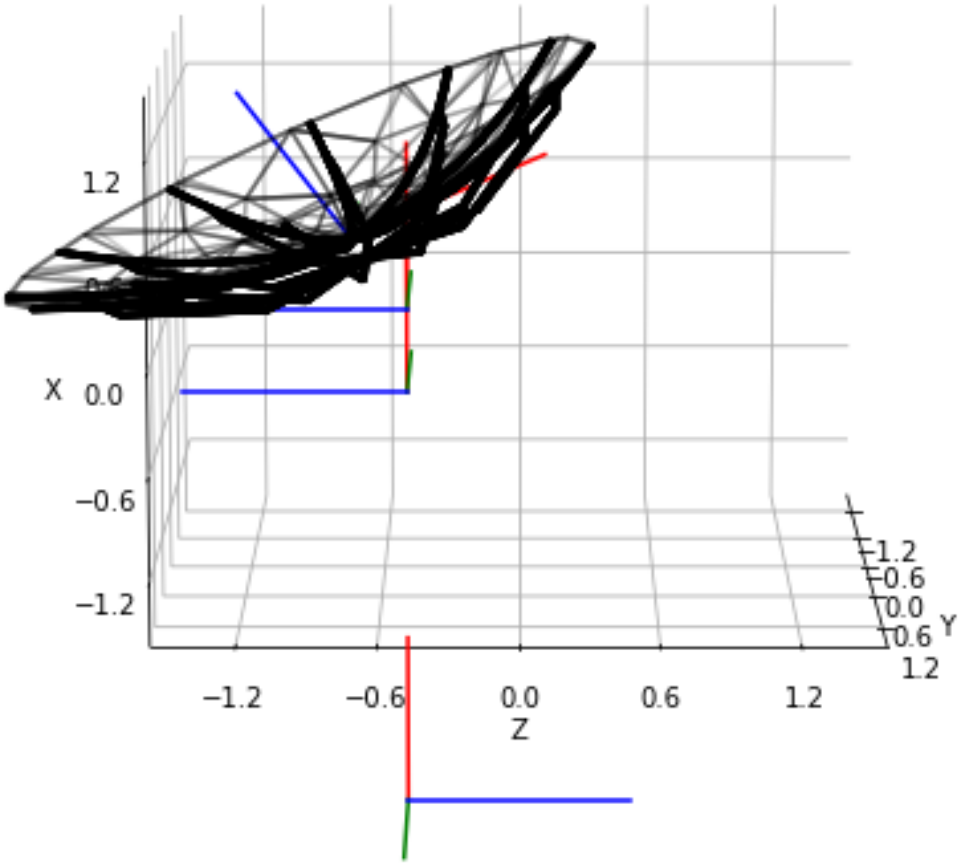
Es decir es un sistema de referencia de cuerpo que rota con la antena pero tiene su origen en el eje de elevación

Peso de la estructura $\mathbf{W = 1715{,}7\ N}$ cuyo centro de gravedad está localizado en el eje de elevación

1.2 Computo dinámico para la selección de los motores

[5]:
```
Write_MaxF(Forces_MaxMy_b,Moments_MaxMy_b,alpha_MaxMy_b,theta_MaxMy_b,"MOMENTO","Y")
```

```
-----------------------------------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN Y
-----------------
 Azimuth:  [180]
 Elevación:  [60]
 Fuerza en x:  -898.5005149333588
 Fuerza en y:  -4.477382141932906
 Fuerza en z:  1519.0291649237538
 Momentos en x:  5.555064284826823
 Momentos en y:  -769.8985014826943
 Momentos en z:  -9.62165358063132
```



Función para el cálculo del tensor de inercia para cualquier posición de la antena

[6]:
```
def Inertia(I_ant,theta,alpha):
    Rcb,T_cb,T_bc,T_ac,T_sc = set_config(alpha,theta,"C")
    Rbc = Rcb.T
    Inertia = np.dot(np.dot(Rbc,I_ant),Rbc.T)
    return Inertia
```

[7]:
```
#Definición Dinámica del sistema de potencia
theta_values = 90 - beta_values
I_ant = [
    [107.663,0,0],
    [0,94.231 ,0],
```

```
    [0,0 ,45.165]
]

I_ant = np.array(I_ant)
W_ant = np.array([-171.57*10,0,0])
W_ant_m = np.empty((len(theta_values),len(search_alpha),3))
W_ant_m[:,:,0] = -171.57*10
```

[8]:
```
#Diseño del sistema para el caso máximo de momento en el eje y



def Forces_xyz(theta_values,search_alpha, Forces_B,theta_config,beta_config):
    points = (theta_values,search_alpha)
    values = Forces_B[:,:]
    config = np.array([theta_config,beta_config])
    forces_b = interpn(points,values,config)
    return forces_b

print(theta_values)
print(search_alpha)
```

```
[ 5 20 40 45 60 75 90]
[  5  20  40  45  60  75  90 120 140 170 180]
```

El siguiente código es sólo sí se quiere ver la confirmación de la interpolación Si se desea realizar la comprobación de lo anterior se debe descomentar, ya sea la prueba gráfica o la prueba númerica

[9]:
```
#Prueba gráfica de la interpolación
# Forces_xyz_b = np.empty((len(beta_values),len(search_alpha),3))
# for i in range(len(beta_values)):
#     for j in range(len(search_alpha)):
#         Forces_xyz_b[i,j] = Forces_xyz(theta_values,search_alpha,␣
 ↪Forces_B,theta_values[i],search_alpha[j])

# Moments_xyz_b = np.empty((len(beta_values),len(search_alpha),3))
# for i in range(len(beta_values)):
#     for j in range(len(search_alpha)):
#         Moments_xyz_b[i,j] = Forces_xyz(theta_values,search_alpha,␣
 ↪Moments_B,theta_values[i],search_alpha[j])


# plot_wrench2(search_alpha,beta_values,Forces_xyz_b,Moments_xyz_b,'rainbow',␣
 ↪"el eje de elevación interpolada")

# Prueba númerica de la interpolación
```

```
# forces_b = Forces_xyz(theta_values,search_alpha,␣
 ↪Forces_B,theta_values[0],search_alpha[10])
# print(Forces_B[0,10])
# print(forces_b)
# print(theta_values[0],"\n",search_alpha[10])
```

[10]:
```
                        #Selección del motor de elevación (eje y)
#Momentos máximos en el eje y de elevación
#Reducción del sistema de potencia
N_plant = 100  #Reducción planetaria
N_gears = 3   #Reduccion de engranajes
Nred = N_plant*N_gears #Reducción total

#Cargas ejercidas por el viente en la posición critica
forces_b = np.abs(Forces_xyz(theta_values,search_alpha, Forces_B,60,180))
moments_b = np.abs(Forces_xyz(theta_values,search_alpha,Moments_B,60,180))


Jload = Inertia(I_ant,60,180) #Matriz de inercia de la carga
alpha_elev = np.array([0, np.pi/18, 0]) #Vector de aceleración, está acelerando␣
 ↪en azimuth y en elevación
omega_elev = np.array([np.pi/18,0, 0])  #Vector de velocidad angular, tiene␣
 ↪velocidad en azimuth y elevación1
omega_skew = pr_rot.cross_product_matrix(omega_elev) #Matriz skewsimetrica para␣
 ↪el computo de producto cruz

#Ecuación dinámica
Moments_acel = np.dot(Jload,alpha_elev) + moments_b + np.dot(omega_skew,np.
 ↪dot(Jload,omega_elev))
print(
    "Moments cálculados en la rotación en el punto critico","\n",
    Moments_acel,"\n",
    "Momentos ejercidos por el viento","\n",
    moments_b
)

#Ecuación dinámica para el motor
Jengr_refl = np.zeros((3,3))
Jmotor = np.array([
    [0,0,0],
    [0, 0.00023 , 0],
    [0,0,0]
])
Jplant = np.array([
    [0,0,0],
    [0, 0.00063, 0],
    [0,0,0]
```

```
])
Jtotal = (1/Nred**2)*Jload + Jengr_refl + Jmotor + Jplant

Moments_acel_m = np.dot(Jtotal,Nred*alpha_elev) + (1/Nred)*Moments_acel
Moments_acel_m
```

```
Moments cálculados en la rotación en el punto critico
 [[  5.55506428 787.16928225   9.62165358]]
 Momentos ejercidos por el viento
 [[  5.55506428 769.89850148   9.62165358]]
```

[10]:
```
array([[0.01851688, 2.72374848, 0.03207218]])
```

1.3 Cargas para sistema de Elevación

1.3.1 Cargas en el acople del eje de elevación con los Brazos del Acople

A continuación se cálculan las cargas en los acoples del eje de elevación basado en las siguientes ecuaciones:

$F_{za} = -\frac{F_{zw}}{2} - \frac{Mx}{L}$

$F_{zb} = -\frac{F_{zw}}{2} + \frac{Mx}{L}$

$F_{xa} = -\frac{F_{xw}}{2} - \frac{Mz}{L}$

$F_{xb} = -\frac{F_{xw}}{2} + \frac{Mz}{L}$

if $(Fyw > 0)$:

$Fya = Fyw$

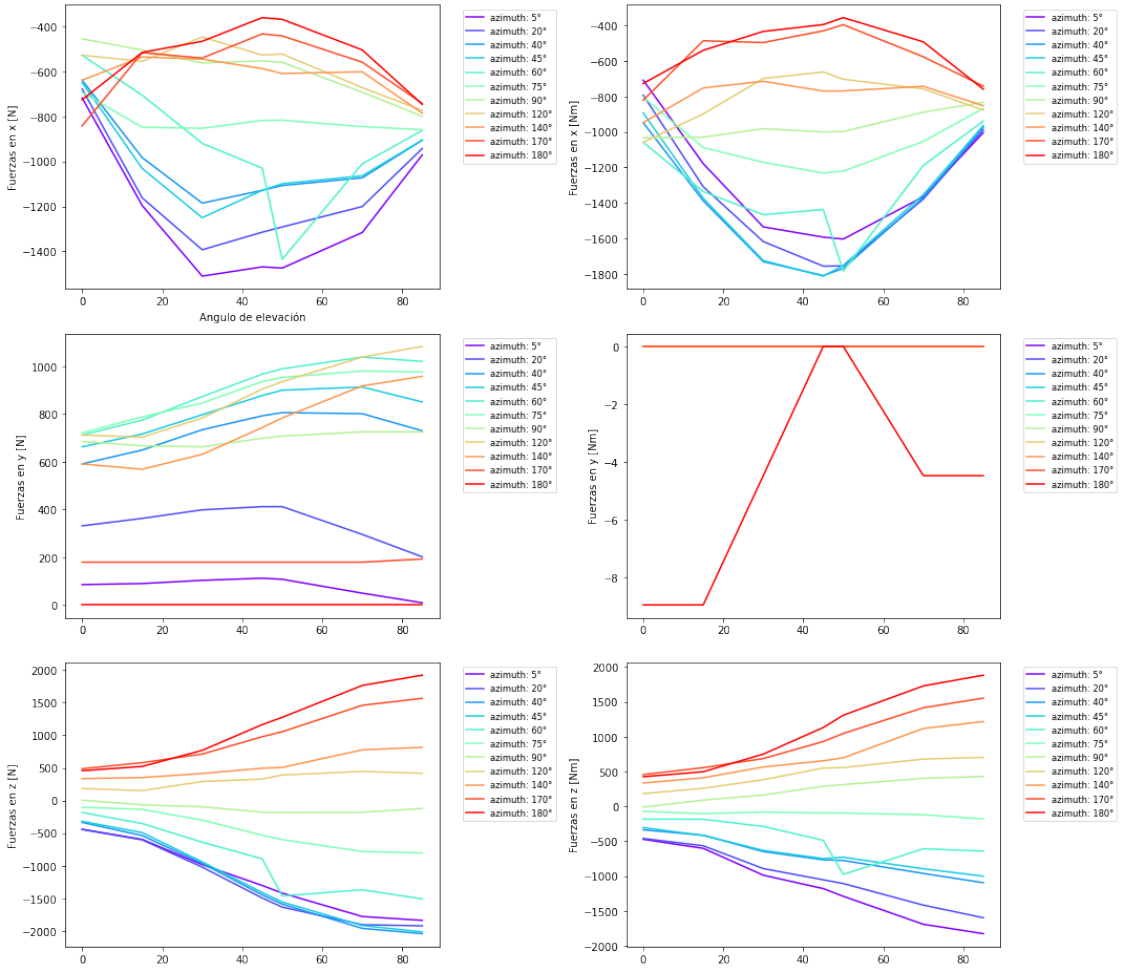
else

$Fyb = Fyw$

[11]:
```
#Distancia entre brazos
L_arm = 0.573

F_a = np.zeros((len(theta_values),len(search_alpha),3))
F_b = np.zeros((len(theta_values),len(search_alpha),3))



#Fuerzas en z
F_a[:,:,2] = (-0.5*Forces_B[:,:,2]) - (1/L_arm)*Moments_B[:,:,0]
F_b[:,:,2] = (-0.5*Forces_B[:,:,2]) + (1/L_arm)*Moments_B[:,:,0]

#Fuerzas en x
F_a[:,:,0] = (-0.5*Forces_B[:,:,0]) - (1/L_arm)*Moments_B[:,:,2]
F_b[:,:,0] = (-0.5*Forces_B[:,:,0]) + (1/L_arm)*Moments_B[:,:,2]
```

```
for i in range(len(theta_values)):
    for j in range(len(search_alpha)):
        if (Forces_B[i,j,1]>0):
            F_a[i,j,1] = -Forces_B[i,j,1]
        else:
            F_b[i,j,1] = -Forces_B[i,j,1]

#Para calcular las reacciones en el eje, se toman las direcciones contrarias a␣
 ↪las cargas en el acople
F_a = -F_a
F_b = -F_b

#Siendo F_a y F_b los vectores de fuerzas en el eje                          ␣
 ↪
```

[12]:
```
plot_forces2(search_alpha,theta_values,F_a,F_b,'rainbow')
```



[13]:
```
Forces_MaxF_a,Moments_MaxF_a,alpha_MaxF_a,theta_MaxF_a= Max_Force(F_a,F_b,2)
Forces_MaxF_a
```

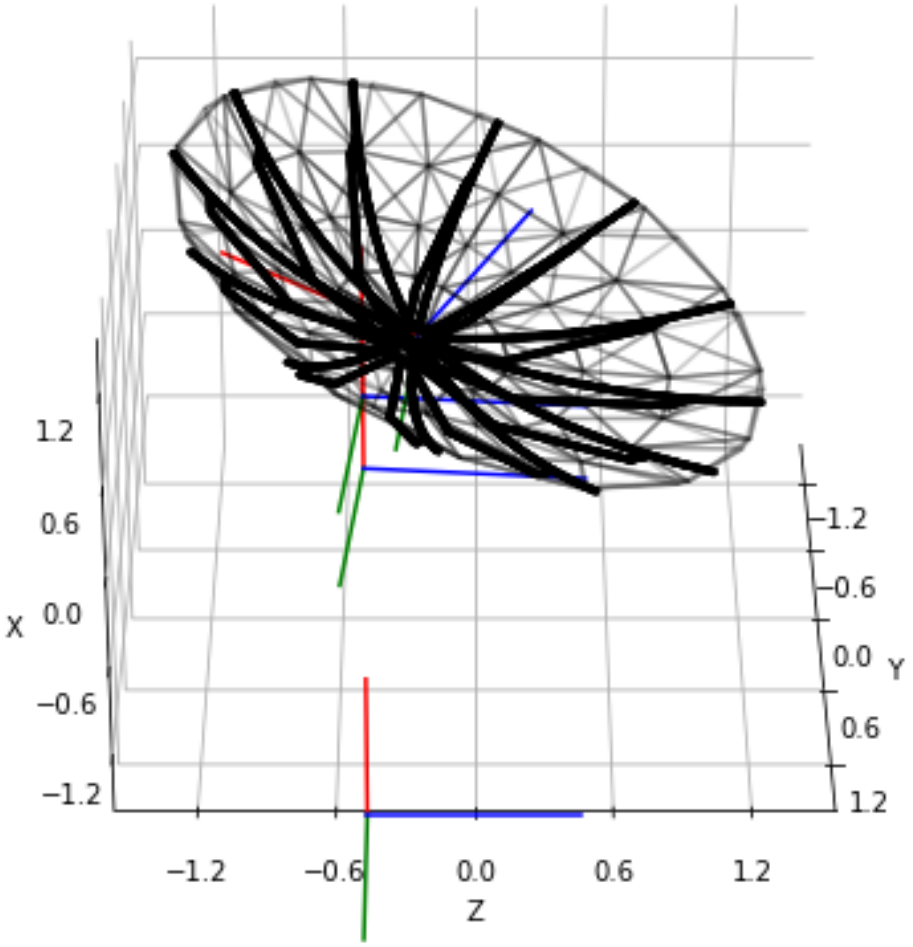
[13]:
```
[array([ -905.59993676,   729.81328914, -2042.49913541])]
```

[14]:
```
#Se reusa las funciones que se plantearon en Radiotelescope_Load_Modeling_V2,␣
 ↪sinedo lo referente a fuerzas = Fa y Momentos Fb
Forces_MaxFx_a,Moments_MaxFx_a,alpha_MaxFx_a,theta_MaxFx_a= Max_Force(F_a,F_b,0)
Forces_MaxFy_a,Moments_MaxFy_a,alpha_MaxFy_a,theta_MaxFy_a= Max_Force(F_a,F_b,1)
Forces_MaxFz_a,Moments_MaxFz_a,alpha_MaxFz_a,theta_MaxFz_a= Max_Force(F_a,F_b,2)

Forces_MaxFx_b,Moments_MaxFx_b,alpha_MaxFx_b,theta_MaxFx_b=␣
 ↪Max_Moments(F_a,F_b,0)
Forces_MaxFy_b,Moments_MaxFy_b,alpha_MaxFy_b,theta_MaxFy_b=␣
 ↪Max_Moments(F_a,F_b,1)
Forces_MaxFz_b,Moments_MaxFz_b,alpha_MaxFz_b,theta_MaxFz_b=␣
 ↪Max_Moments(F_a,F_b,2)
```

[15]:
```
def Write_MaxF1(Forces_Max,Moments_Max,alpha_max,theta_max,carga,X):
    print(
"-----------------------------------ESTADO DE CARGA PARA LA MÁXIMA",␣
 ↪carga,"EN", X, "-----------------","\n",
"Azimuth: ",alpha_max,"\n",
"Elevación: ", theta_max,"\n",
"Fuerza en A en x: ",Forces_Max[0][0],"\n",
"Fuerza en A en y: ",Forces_Max[0][1],"\n",
"Fuerza en A en z: ",Forces_Max[0][2],"\n",
"Fuerza en B en x: ",Moments_Max[0][0],"\n",
"Fuerza en B en y: ",Moments_Max[0][1],"\n",
"Fuerza en B en z: ",Moments_Max[0][2],"\n",
    )
    plot_config(alpha_max[0],theta_max[0],0,30,"SI")
```

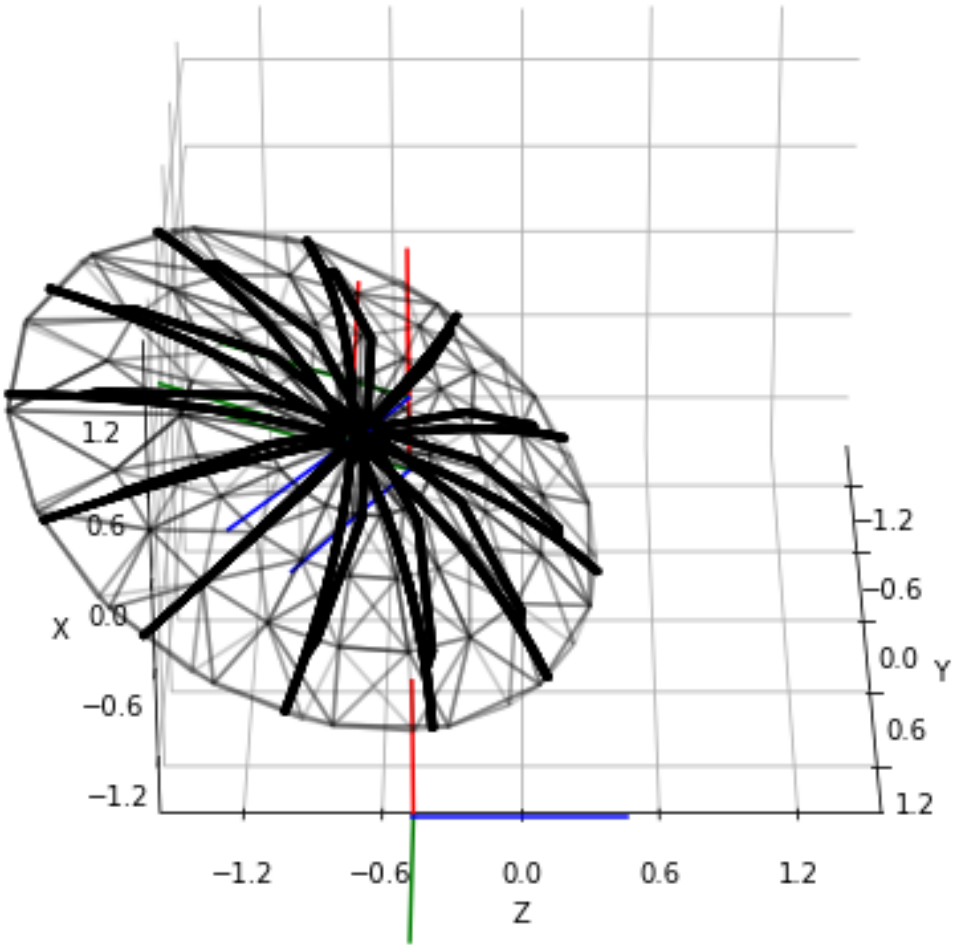
[16]:
```
Write_MaxF1(Forces_MaxFx_a,Moments_MaxFx_a,alpha_MaxFx_a,theta_MaxFx_a,"FUERZA␣
 ↪EN A","X")
Write_MaxF1(Forces_MaxFy_a,Moments_MaxFy_a,alpha_MaxFy_a,theta_MaxFy_a,"FUERZAEN␣
 ↪A","Y")
Write_MaxF1(Forces_MaxFz_a,Moments_MaxFz_a,alpha_MaxFz_a,theta_MaxFz_a,"FUERZA␣
 ↪EN A","Z")
Write_MaxF1(Forces_MaxFx_b,Moments_MaxFx_b,alpha_MaxFx_b,theta_MaxFx_b,"FUERZA␣
 ↪EN B","X")
Write_MaxF1(Forces_MaxFy_b,Moments_MaxFy_b,alpha_MaxFy_b,theta_MaxFy_b,"FUERZA␣
 ↪EN B","Y")
Write_MaxF1(Forces_MaxFz_b,Moments_MaxFz_b,alpha_MaxFz_b,theta_MaxFz_b,"FUERZA␣
 ↪EN B","Z")
```

----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN A EN X ----------
Azimuth: [5]
Elevación: [60]
Fuerza en A en x: -1510.4513053402482
Fuerza en A en y: 102.97978926445681
Fuerza en A en z: -983.1999718309256
Fuerza en B en x: -1535.6275690012696
Fuerza en B en y: -0.0
Fuerza en B en z: -986.6412800269514



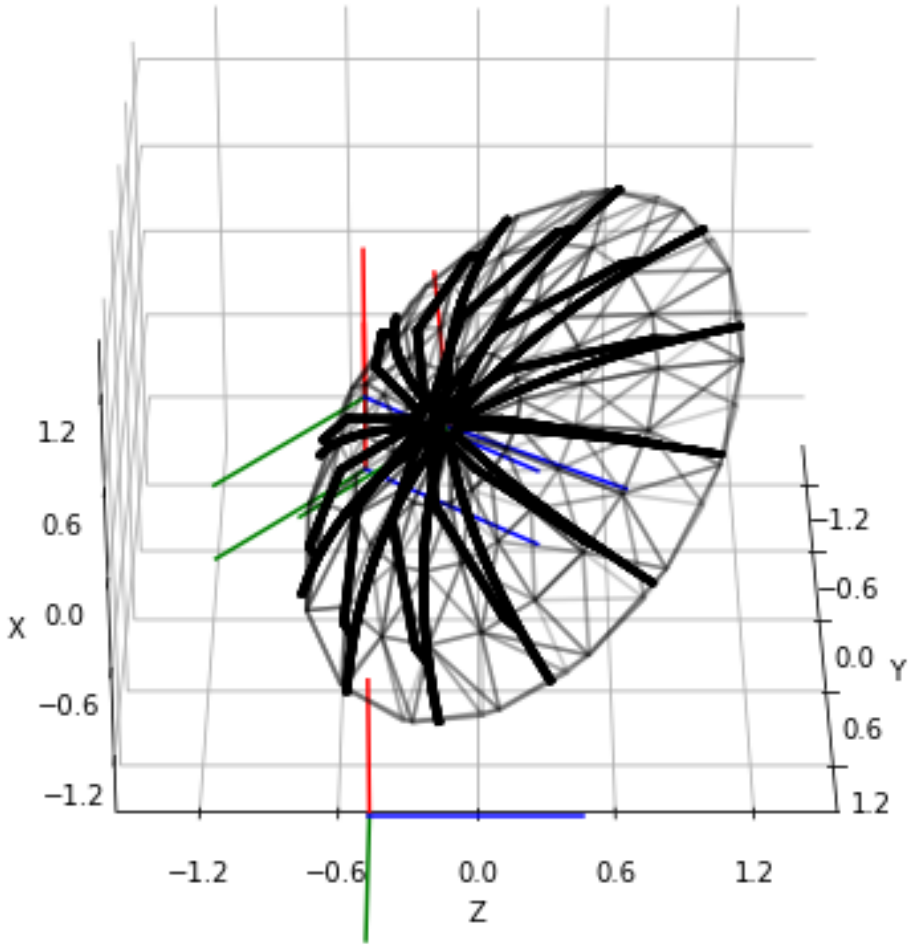
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZAEN A EN Y ----------

Azimuth: [120]
Elevación: [5]
Fuerza en A en x: -773.3333245094051
Fuerza en A en y: 1083.526478347763
Fuerza en A en z: 418.34835812678654
Fuerza en B en x: -876.3747935722669
Fuerza en B en y: -0.0
Fuerza en B en z: 702.9814027125935



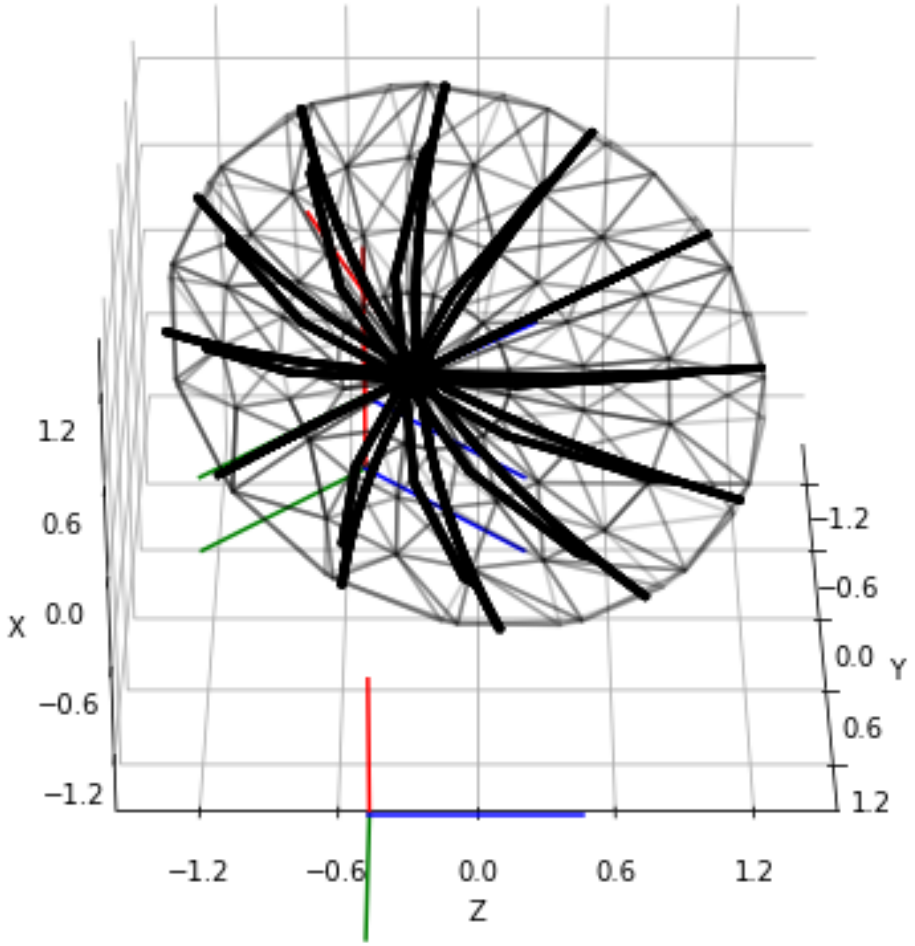
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN A EN Z ----------
Azimuth: [40]
Elevación: [5]

Fuerza en A en x: -905.5999367602299
Fuerza en A en y: 729.8132891350637
Fuerza en A en z: -2042.499135412641
Fuerza en B en x: -972.8757090557569
Fuerza en B en y: -0.0
Fuerza en B en z: -1094.9062721348882



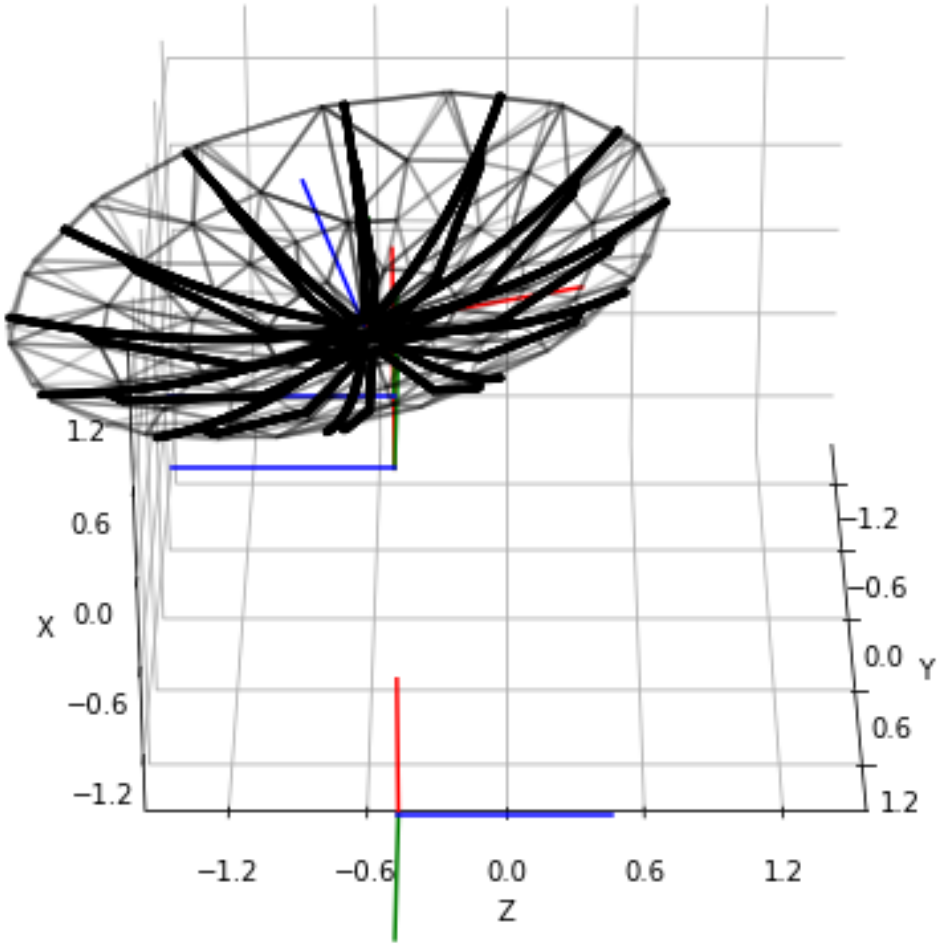
----------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN B EN X ----------
Azimuth: [45]
Elevación: [45]
Fuerza en A en x: -1128.7269779801359
Fuerza en A en y: 877.5668998188495

```
 Fuerza en A en z:  -1411.0136218757098
 Fuerza en B en x:  -1813.5679318295875
 Fuerza en B en y:  -0.0
 Fuerza en B en z:  -748.1896993498675
```

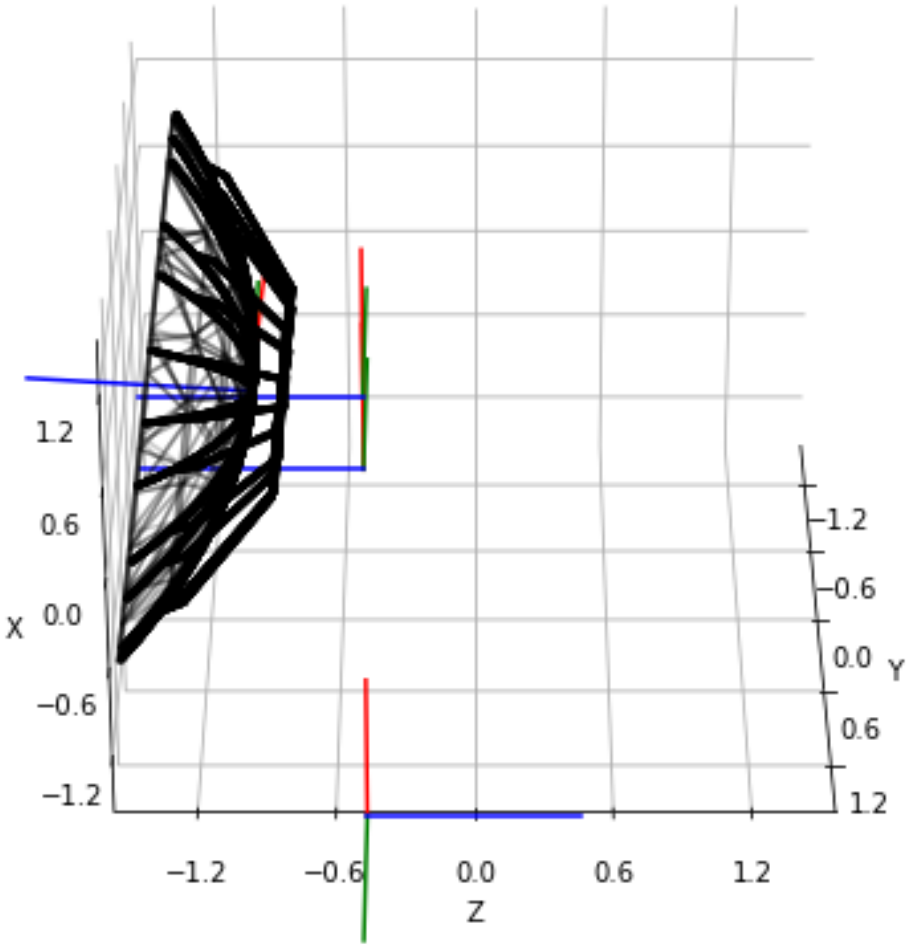



```
-----------------------------------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN B
EN Y -----------------
 Azimuth:  [180, 180]
 Elevación:  [75, 90]
 Fuerza en A en x:  -515.5318049165809
 Fuerza en A en y:  -0.0
 Fuerza en A en z:  522.6970350102949
 Fuerza en B en x:  -538.9031651654203
```

```
 Fuerza en B en y:  -8.954764283865812
 Fuerza en B en z:  496.72420099355315
```




```
-----------------------------------ESTADO DE CARGA PARA LA MÁXIMA FUERZA EN B
EN Z -----------------
 Azimuth:  [180]
 Elevación:  [5]
 Fuerza en A en x:  -746.3239557682843
 Fuerza en A en y:  -0.0
 Fuerza en A en z:  1922.6645404080066
 Fuerza en B en x:  -758.4533096487146
 Fuerza en B en y:  -4.477382141932906
 Fuerza en B en z:  1882.6764224288124
```



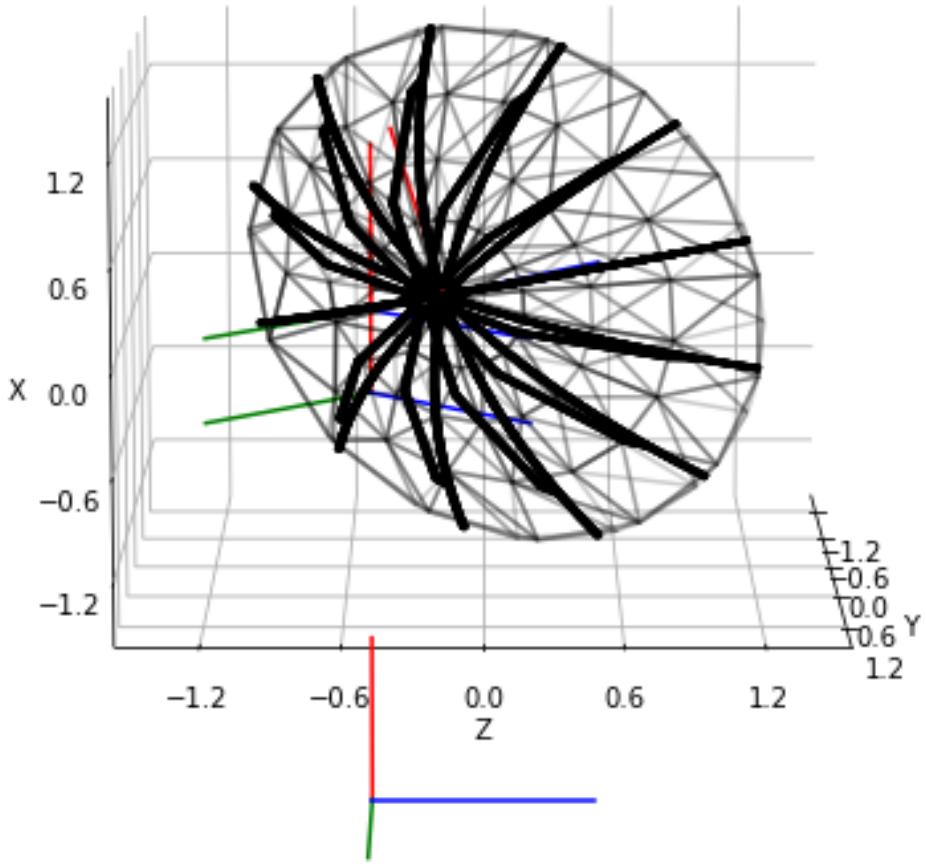
1.4 Calculo de Torques para el Eje del azimut

A continuación se presenta el cálculo para dimensionamiento de los motores en el eje del azimut para la antena propuesta, para esto se necesita de la matriz de Inercia de todo el sistema de elevación en su totalidad, para el diseño propuesto se tiene una matriz de Inercia.

[17]:
```
Write_MaxF(Forces_MaxMx_a,Moments_MaxMx_a,alpha_MaxMx_a,theta_MaxMx_a,"MOMENTO","X")
```

```
-----------------------------------ESTADO DE CARGA PARA LA MÁXIMA MOMENTO EN X
-----------------
 Azimuth:  [45]
 Elevación:  [20]
```

```
Fuerza en x:  -3368.452194000706
Fuerza en y:  913.3859569543126
Fuerza en z:  -2812.513772643336
Momentos en x:  -293.21405189721685
Momentos en y:  920.5459875915878
Momentos en z:  316.81499555920203
```



[18]:
```
# "Sistema de Potencia de Elevación"
m_elev = 99.576
x_cg_az = -0.2468
y_cg_az = 0
z_cg_az = 0
```

```
X_cg = [x_cg_az,y_cg_az,z_cg_az]
I_ant_elev = np.array([
    [4.058,0,0],
    [0,3.144,0],
    [0,0,2.430]
])


I_ant_elev[0,0] = I_ant_elev[0,0] + m_elev*(X_cg[1]**2 + X_cg[2]**2)
I_ant_elev[1,1] = I_ant_elev[1,1] + m_elev*(X_cg[0]**2 + X_cg[2]**2)
I_ant_elev[2,2] = I_ant_elev[2,2] + m_elev*(X_cg[0]**2 + X_cg[1]**2)

# "Inercia Asociada a la Antena"
m_antena = 171.57
I_antena = Inertia(I_ant,40,20)

I_antena[0,0] = I_antena[0,0] + m_antena*(X_cg[1]**2 + X_cg[2]**2)
I_antena[1,1] = I_antena[1,1] + m_antena*(X_cg[0]**2 + X_cg[2]**2)
I_antena[2,2] = I_antena[2,2] + m_antena*(X_cg[0]**2 + X_cg[1]**2)

I_sistema_pot = I_antena + I_ant_elev #Inercia de la carga

# "Definición del sistema de transimisión"

#Reducción del sistema de potencia
N_plant = 100  #Reducción planetaria
N_gears = 3   #Reduccion de engranajes
Nred = N_plant*N_gears #Reducción total

#Cargas ejercidas por el viente en la posición critica
forces_a = np.abs(Forces_xyz(theta_values,search_alpha, Forces_A,40,20))
moments_a = np.abs(Forces_xyz(theta_values,search_alpha,Moments_A,40,20))



alpha_az = np.array([np.pi/18, 0, 0]) #Vector de aceleración, está acelerando␣
 ↪en azimuth y en elevación
omega_az = np.array([0,0, 0])  #Vector de velocidad angular, tiene velocidad en␣
 ↪azimuth y elevación1
omega_skew_az = pr_rot.cross_product_matrix(omega_az) #Matriz skewsimetrica␣
 ↪para el computo de producto cruz

#Ecuación dinámica
Moments_acel_az = np.dot(I_sistema_pot,alpha_az) + moments_a + np.
 ↪dot(omega_skew_az,np.dot(I_sistema_pot,omega_az))
print(
    "Moments cálculados en la rotación en el punto critico","\n",
```

```
    Moments_acel_az,"\n",
    "Momentos ejercidos por el viento","\n",
    moments_a
)
J_plan = 0.00063
I_rotor = 0.00023
Inertia_ratio_az = (((1/Nred**2)*I_sistema_pot[1,1]) + J_plan)/I_rotor

print(
    "La razon de inercia es de:", Inertia_ratio_az
)
```

```
Moments cálculados en la rotación en el punto critico
 [[ 166.4281915  1143.71426492  232.97272837]]
 Momentos ejercidos por el viento
 [[ 151.43610436 1143.71426492  238.34384955]]
La razon de inercia es de: 8.241090238407729
```

1.4.1 Calculo del Tilt Moment para la selección del Slew Bearing

En las siguientes lineas de código se evalua la condición de carga para el Slew Bearing, mostrando los momentos de cabeceo, fuerza axial y radial que soportaria, para ello se evalua cuando el Mz es máximo, debido a que el momento en y está soportado por el sistema de potencia de elevación

En la presentación de máximas cargas radial, axial y de momento de inclinación la convención es

[Momento de inclinación] [Fuerza axial] [Fuerza radial]

[19]:
```
#Calculo de momentos

I_antena = Inertia(I_ant,5,180)

I_antena[0,0] = I_antena[0,0] + m_antena*(X_cg[1]**2 + X_cg[2]**2)
I_antena[1,1] = I_antena[1,1] + m_antena*(X_cg[0]**2 + X_cg[2]**2)
I_antena[2,2] = I_antena[2,2] + m_antena*(X_cg[0]**2 + X_cg[1]**2)

I_sistema_pot = I_antena + I_ant_elev #Inercia de la carga

M_y2 = Moments_A[:,:,1]
M_y2 = M_y2**2

M_z2 = Moments_A[:,:,2]
M_z2 = M_z2**2

M_tilt = (M_y2+M_z2)**(1/2)
Max_M = np.amax(M_tilt[:,:])
Max_M_idx = np.where(M_tilt[:,:] == Max_M)
M_tilt[Max_M_idx]
```

```
F_y2 = Forces_A[:,:,1]
F_y2 = F_y2**2

F_z2 = Forces_A[:,:,2]
F_z2 = F_z2**2

#Definicion de las fuerzas en el slew
F_axial = Forces_A[:,:,0]

F_radial = (F_y2 + F_z2)**(1/2)


#Calculo de las situaciones máximas
Max_F_axial = np.amax(F_axial[:,:])
Max_F_axial_idx = np.where(F_axial[:,:] == Max_F_axial)
Max_F_radial = np.amax(F_radial[:,:])
Max_F_radial_idx = np.where(F_radial[:,:] == Max_F_radial)


#Calculo del estado de cargas

Max_M_tilt = np.
 ↪array([M_tilt[Max_M_idx],F_axial[Max_M_idx],F_radial[Max_M_idx]])
Max_Fr = np.
 ↪array([M_tilt[Max_F_radial_idx],F_axial[Max_F_radial_idx],F_radial[Max_F_radial_idx]])
Max_Fax = np.
 ↪array([M_tilt[Max_F_axial_idx],F_axial[Max_F_axial_idx],F_radial[Max_F_axial_idx]])

print("El estado de cargas para momento de inclinación máximo:␣
 ↪","\n",Max_M_tilt)
print("El estado de cargas para momento de fuerza radial máxima: ","\n",Max_Fr)
print("El estado de cargas para momento de fuerza radial máxima: ","\n",Max_Fax)
```

```
El estado de cargas para momento de inclinación máximo:
 [[ 1362.36585788]
 [-1672.94839384]
 [ 2584.52207688]]
El estado de cargas para momento de fuerza radial máxima:
 [[  883.49710574]
 [-2454.77726542]
 [ 3805.34359689]]
El estado de cargas para momento de fuerza radial máxima:
 [[ 1362.36585788]
 [-1672.94839384]
 [ 2584.52207688]]
```

[20]:
```
# Determinación de la fuerza Axial equivalente para la selección del rodamiento
f_l = 5
Sel_Tilt = np.array([Max_M_tilt[0],f_l*(Max_M_tilt[1] + 2.3*Max_M_tilt[2])])
Sel_Fr = np.array([Max_Fr[0],f_l*(Max_Fr[1] + 2.3*Max_Fr[2])])
Sel_Fa = np.array([Max_Fax[0],f_l*(Max_Fax[1] + 2.3*Max_Fax[2])])

print("Selección por momento de inclinación: ","\n", Sel_Tilt)
print("Selección por carga axial: ","\n", Sel_Fa)
print("Selección por carga radial: ","\n", Sel_Fr)
```

```
Selección por momento de inclinación:
 [[ 1362.36585788]
 [21357.26191489]]
Selección por carga axial:
 [[ 1362.36585788]
 [21357.26191489]]
Selección por carga radial:
 [[  883.49710574]
 [31487.56503715]]
```