# N06. Presupuesto de compra de energía 2022 - 2028

En este documento se elaborará el presupuesto estimado para la compra de energía del Bloque Carara, durante el periodo comprendido entre diciembre de 2022 y diciembre de 2028.

Se asume que la oferta de ENEL\_M2 fue la alternativa seleccionada. El presupuesto se calculará con base en los escenarios de precios previamente proyectados para los 78 meses del análisis.

El presupuesto por compra de energía se refiere a valores económicos y no financieros, lo que significa que no tienen factores de descuento.

Inicialmente se analizarán los egresos por cargos No Regulados. Luego se proyectarán los componentes Regulados y se calcularán los egresos por cargos Regulados. Por último se sumarán los egresos No Regulados más los egresos Regulados.

## 6.1. Egresos por concepto de los cargos No Regulados por periodos

Se tomarán como base los datos elaborados en la selección de la oferta.

### 6.1.1. Lee los egresos mensuales de cargos No Regulados por escenario

Los valores de egresos mensuales están expresados en miles de millones de pesos.

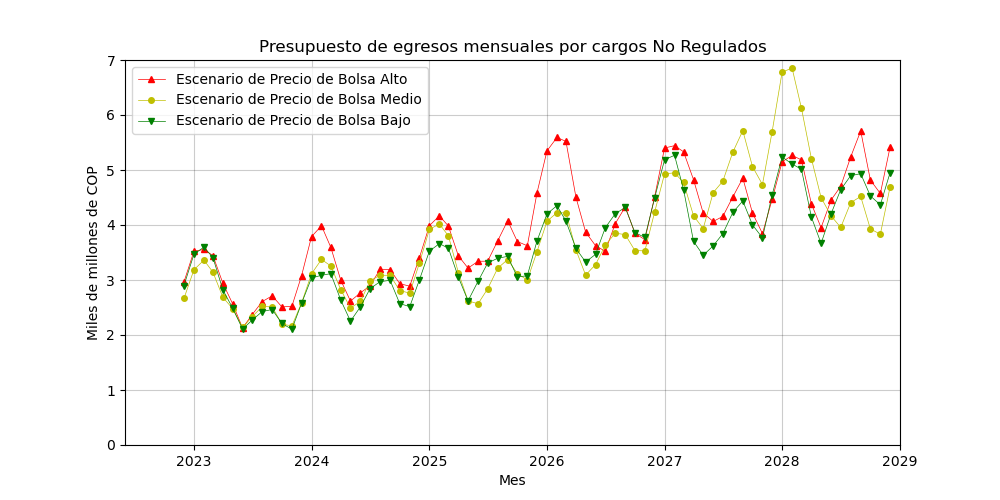
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.1.1. Carga paquetes requeridos para elaborar el presupuesto.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
import numpy as np  
import pandas as pd  
import datetime as dt  
import matplotlib.pyplot as plt  
import matplotlib.ticker as ticker  
import matplotlib as mpl  
import os  
%matplotlib widget

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.1.2. Lee los egresos\_ENEL\_M2.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_ENEL\_M2 = pd.read\_parquet('parquet/egreso\_ENEL\_M2.parquet.gzip', \  
 engine='fastparquet')  
egreso\_ENEL\_m = egreso\_ENEL\_M2[['egreso\_alto', 'egreso\_medio', 'egreso\_bajo']]   
#Millones de COP. Egresos mensuales de ENEL en 3 escenarios.  
egreso\_ENEL\_m

egreso\_alto egreso\_medio egreso\_bajo  
index   
2022-12-01 2963.579154 2664.415876 2898.801840  
2023-01-01 3522.481168 3185.495330 3467.627803  
2023-02-01 3555.816984 3361.517087 3597.934446  
2023-03-01 3443.051227 3142.729552 3407.272083  
2023-04-01 2938.577331 2683.973015 2819.191931  
... ... ... ...  
2028-08-01 5238.355205 4406.496114 4897.531684  
2028-09-01 5710.646936 4518.717844 4932.677330  
2028-10-01 4817.596144 3928.578689 4528.795419  
2028-11-01 4572.969366 3832.626111 4366.322025  
2028-12-01 5410.393353 4687.700753 4948.473906  
  
[73 rows x 3 columns]

### 6.1.2. Presupuesto mensual de cargos No Regulados por escenario

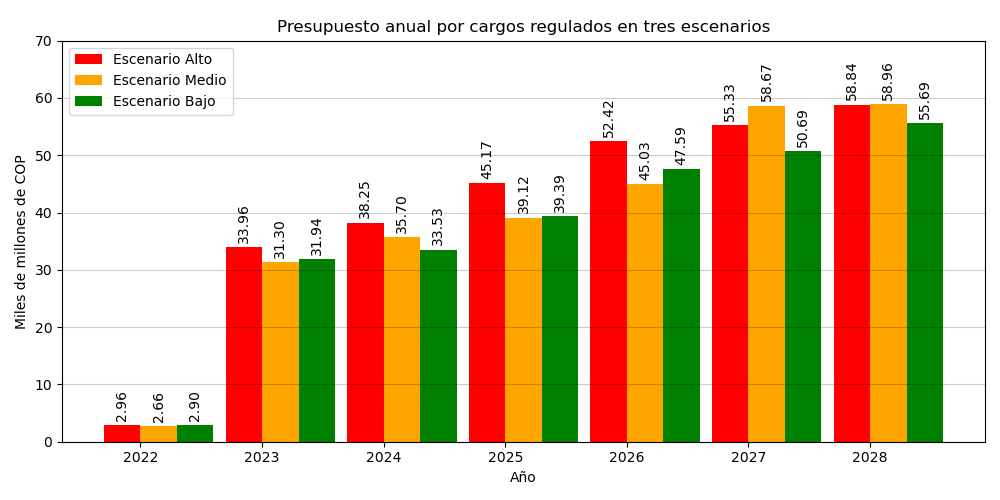
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.1.3. Grafica egresos mensuales, egreso\_ENEL\_m.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Presupuesto de egresos mensuales por cargos No Regulados')  
ax1.set\_ylabel ('Miles de millones de COP')  
ax1.set\_xlabel ('Mes')  
# ax1.plot(pbm\_py.index, pbm\_py.pbm, 'b-', \  
# label='Precio de Bolsa real', markersize=2)  
ax1.plot(egreso\_ENEL\_m.index, (egreso\_ENEL\_m.egreso\_alto)/1e3, 'r-^', \  
 label='Escenario de Precio de Bolsa Alto', linewidth= 0.5, markersize=4)  
ax1.plot(egreso\_ENEL\_m.index, (egreso\_ENEL\_m.egreso\_medio)/1e3, 'y-o', \  
 label='Escenario de Precio de Bolsa Medio', linewidth= 0.5, markersize=4)  
ax1.plot(egreso\_ENEL\_m.index, (egreso\_ENEL\_m.egreso\_bajo)/1e3, 'g-v', \  
 label='Escenario de Precio de Bolsa Bajo', linewidth= 0.5, markersize=4)  
# ax1.axvspan(pd.to\_datetime('2022-07-01'), pd.to\_datetime('2029-01-01'), \  
# color='#808080', alpha=0.3)  
plt.legend(loc='best')  
plt.xlim(pd.to\_datetime('2022-06-01'), pd.to\_datetime('2029-01-01'))  
plt.ylim(0, 7)  
plt.grid(axis='both', color='k', alpha=0.2)  
  
y\_labels = ax1.get\_yticks()  
ax1.yaxis.set\_major\_formatter(mpl.ticker.StrMethodFormatter('{x:,.0f}'))  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.1.4. Calcula los egresos anuales, egreso\_ENEL\_a.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_ENEL\_a = egreso\_ENEL\_m.groupby(egreso\_ENEL\_m.index.to\_period('Y')).sum()

### 6.1.3. Presupuesto anual de cargos No Regulados por escenario

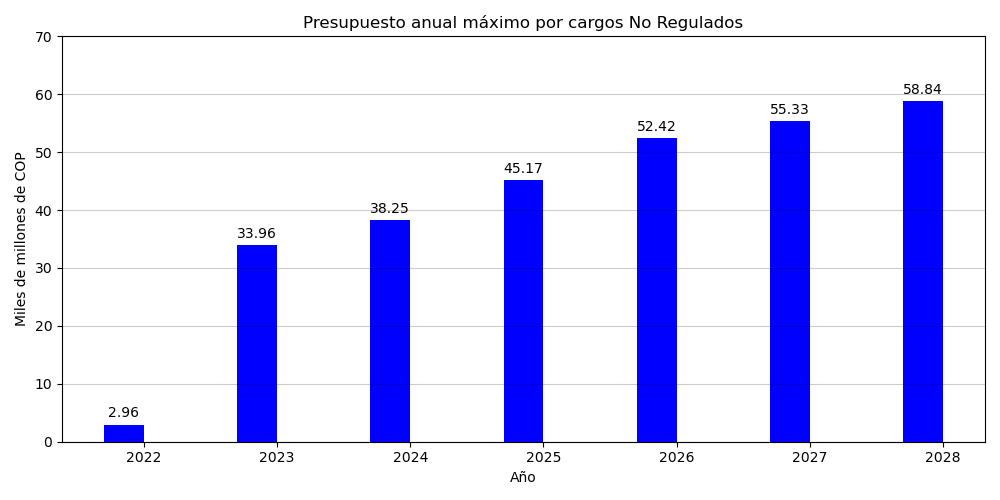
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.5. Grafica el egreso anual por cargos No Regulados en tres escenarios.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
anios = egreso\_ENEL\_a.index  
escenario\_alto = ((egreso\_ENEL\_a.egreso\_alto)/1e3).round(2)  
escenario\_medio = ((egreso\_ENEL\_a.egreso\_medio)/1e3).round(2)  
escenario\_bajo = ((egreso\_ENEL\_a.egreso\_bajo)/1e3).round(2)  
  
  
x = np.arange(len(escenario\_alto)) # La localización de etiquetas.  
width = 0.30 # El ancho de las barras.  
  
rects1 = ax.bar(x - width/2, escenario\_alto, width, label='Escenario Alto', \  
 color='r')  
rects2 = ax.bar(x + width/2, escenario\_medio, width, label='Escenario Medio', \  
 color='orange')  
rects3 = ax.bar(x + 3 \* width/2, escenario\_bajo, width, label='Escenario Bajo', \  
 color='g')  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto anual por cargos regulados en tres escenarios')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
ax.set\_xticks(x, anios)  
ax.legend(loc=2)  
  
ax.bar\_label(rects1, padding=3, fmt='%0.2f', rotation=90)  
ax.bar\_label(rects2, padding=3, fmt='%0.2f', rotation=90)  
ax.bar\_label(rects3, padding=3, fmt='%0.2f', rotation=90)  
  
fig.tight\_layout()  
  
plt.ylim(0, 70)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



### 6.1.4. Presupuesto anual máximo de cargos No Regulados

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.6. Calcula el egreso máximo por cargos regulados por cada año.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_ENEL\_a\_max = egreso\_ENEL\_a.max(axis=1)

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.7. Grafica el máximo egreso anual por cargos No Regulados.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
anios = egreso\_ENEL\_a\_max.index  
maximo\_egreso\_anual = ((egreso\_ENEL\_a\_max)/1e3).round(2)  
  
  
x = np.arange(len(escenario\_alto)) # La localización de etiquetas.  
width = 0.30 # El ancho de las barras.  
  
rects1 = ax.bar(x - width/2, escenario\_alto, width, label='Escenario Alto', \  
 color='b')  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto anual máximo por cargos No Regulados')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
ax.set\_xticks(x, anios)  
# ax.legend(loc=2)  
  
ax.bar\_label(rects1, padding=3, fmt='%0.2f', rotation=0)  
  
fig.tight\_layout()  
  
plt.ylim(0, 70)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



## 6.2. Egresos por concepto de los cargos Regulados por periodos

Se tomarán como base histórica para la proyección, los cargos regulados que aparecen facturados en las liquidaciones por ECOPETROL Energía.

### 6.2.1. Lee los cargos Regulados históricos

Los valores de los cargos se extraen de la hoja 'Liquidación' de los libros Excel 'Bitacora\_Factura\_Frontera\_Frt11545 enero 2022.xlsx', los cuales son suministrados por ECOPETROL Energía a CEPSA, para los meses de octubre de 2021 a abril de 2022.

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.1. Lee lista de todos los libros Excel disponibles de 'Bitacora\_Factura'.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
lista\_bitacoras = os.listdir('bitacoras\_consumos')  
df\_bitacoras = pd.DataFrame({'nombre\_archivo': lista\_bitacoras,   
 'mes': np.NaN, 'stn': np.NaN, 'str': np.NaN, 'restricciones': np.NaN,   
 'ocv': np.NaN, 'perdidas': np.NaN}, index=range(0, len(lista\_bitacoras)))  
df\_bitacoras.sort\_index(axis=0, inplace=True)  
df\_bitacoras.loc[:, 'mes'] = pd.to\_datetime(  
 df\_bitacoras.nombre\_archivo.str.slice(-11, -7) + '-' +   
 df\_bitacoras.nombre\_archivo.str.slice(-7, -5) + '-01')

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.2. Lee los valores de los cargos regulados.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
for i, nombre in enumerate(df\_bitacoras.nombre\_archivo, start=0):   
 if nombre[-11 : -7] == '2019':  
 libro = pd.read\_excel('bitacoras\_consumos/' + nombre,   
 sheet\_name='Liquidación', skiprows=94, usecols="B:Z")  
 df\_bitacoras.loc[i, 'stn'] = libro.iloc[2, 1:25].mean()  
 df\_bitacoras.loc[i, 'str'] = libro.iloc[43, 1]  
 df\_bitacoras.loc[i, 'restricciones'] = libro.iloc[84, 1] - 1  
 df\_bitacoras.loc[i, 'ocv'] = 1  
 df\_bitacoras.loc[i, 'perdidas'] = libro.iloc[125, 1] + \  
 libro.iloc[166, 1:25].mean()  
 elif nombre[-11 : -7] == '2020':  
 libro = pd.read\_excel('bitacoras\_consumos/' + nombre,   
 sheet\_name='Liquidación', skiprows=90, usecols="B:Z")  
 df\_bitacoras.loc[i, 'stn'] = libro.iloc[2, 1:25].mean()  
 df\_bitacoras.loc[i, 'str'] = libro.iloc[43, 1]  
 df\_bitacoras.loc[i, 'restricciones'] = libro.iloc[84, 1] - 1.0  
 df\_bitacoras.loc[i, 'ocv'] = 1.0  
 df\_bitacoras.loc[i, 'perdidas'] = libro.iloc[125, 1] + \  
 libro.iloc[166, 1:25].mean()  
 else:  
 libro = pd.read\_excel('bitacoras\_consumos/' + nombre,   
 sheet\_name='Liquidación', skiprows=90, usecols="B:Z")  
 df\_bitacoras.loc[i, 'stn'] = libro.iloc[2, 1:25].mean()  
 df\_bitacoras.loc[i, 'str'] = libro.iloc[43, 1]  
 df\_bitacoras.loc[i, 'restricciones'] = libro.iloc[84, 1]  
 df\_bitacoras.loc[i, 'ocv'] = libro.iloc[127, 1]  
 df\_bitacoras.loc[i, 'perdidas'] = libro.iloc[170, 1:25].mean()  
# df\_bitacoras

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.3. Salva y lee df\_bitacoras.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# # # #Salva df\_bitacoras  
df\_bitacoras.to\_parquet\  
 ('parquet/df\_bitacoras.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
df\_bitacoras.to\_excel('xlsx/df\_bitacoras.xlsx', sheet\_name='hoja\_1')  
#Lee df\_bitacoras  
df\_bitacoras = pd.read\_parquet\  
 ('parquet/df\_bitacoras.parquet.gzip', engine='fastparquet')

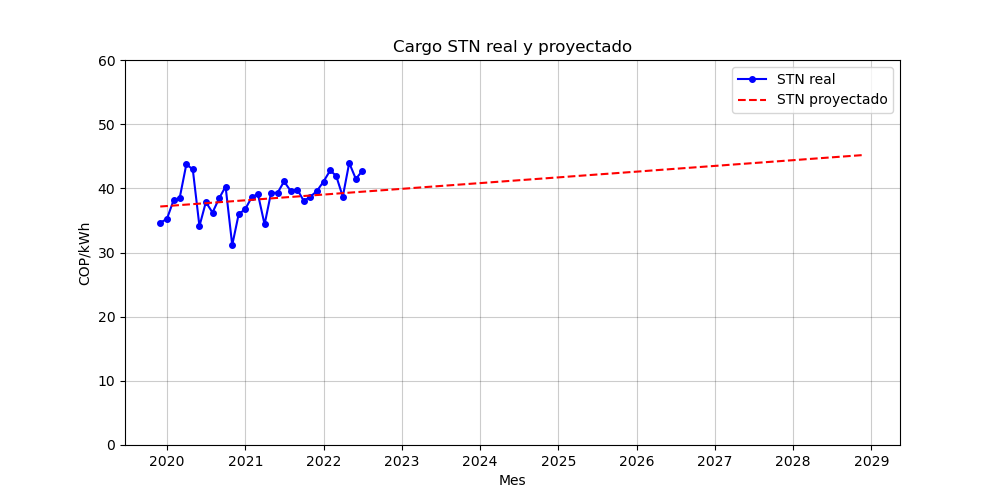
### 6.2.2. Proyecta los cargos regulados

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.4. Crea la estructura de cargos\_regu para la proyección.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
cargos\_regu = df\_bitacoras.copy()  
cargos\_regu.drop(columns='nombre\_archivo', axis=1, inplace=True)  
cargos\_regu.set\_index('mes', inplace=True)  
# Agrega mayo y junio de PDF.  
# ('2022-05-01'). Transmisión: 43.96; Distribución: 35.34; Restricciones 47.58;   
# Otros Cargos: 1.16; Pérdidas: 23.26.  
# ('2022-06-01'). Transmisión: 41.46; Distribución: 33.64; Restricciones 46.75;   
# Otros Cargos 1.21; Pérdidas:23.56.  
cargos\_regu.loc[pd.to\_datetime('2022-05-01')] = [43.96, 35.34, 47.58, 1.16, 23.26]  
cargos\_regu.loc[pd.to\_datetime('2022-06-01')] = [41.46, 33.64, 46.75, 1.21, 23.56]  
cargos\_regu.loc[pd.to\_datetime('2021-09-01')] = [39.80, 26.55, 28.43, 1.01, 5.90]  
cargos\_regu = cargos\_regu.sort\_index()  
# Agrega n\_mes  
cargos\_regu.insert(loc=len(cargos\_regu.columns), column='n\_mes',   
 value=range(0, len(cargos\_regu)))  
# Agrega los meses a proyectar y las columnas de cargos proyectadas, '\_hat'.  
inicio\_proyeccion = '2022-08-01'  
fin\_proyeccion = '2028-12-01'  
longitud\_proyeccion = len(pd.date\_range(start=inicio\_proyeccion,   
 end=fin\_proyeccion, freq='MS'))  
meses\_py = pd.DataFrame({'stn': np.NaN, 'str':np.NaN, 'restricciones':np.NaN,   
 'ocv':np.NaN, 'perdidas':np.NaN, 'n\_mes':range(len(cargos\_regu),   
 len(cargos\_regu) + longitud\_proyeccion), 'stn\_hat': np.NaN,   
 'str\_hat':np.NaN, 'restricciones\_hat':np.NaN,   
 'ocv\_hat':np.NaN, 'perdidas\_hat':np.NaN,},   
 index=pd.date\_range(start=inicio\_proyeccion, end=fin\_proyeccion, freq='MS'))  
cargos\_regu = pd.concat([cargos\_regu, meses\_py])

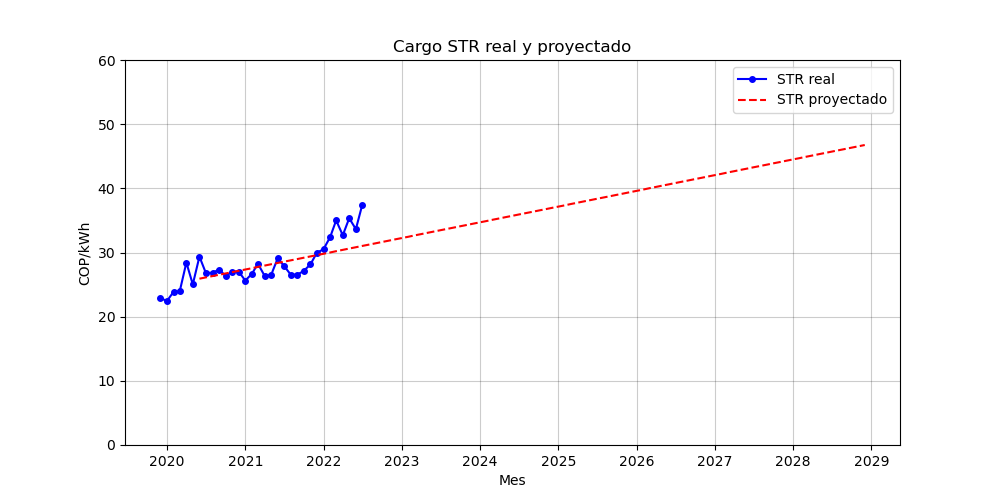
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.5. Elabora la proyección de los cargos\_nr.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
inicio\_ajuste = '2019-12-01'  
final\_ajuste = '2021-12-01'  
inicio\_proyeccion = '2019-12-01'  
final\_proyeccion = '2028-12-01'  
  
#STN  
m, b = np.polyfit(cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'n\_mes'],   
cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'stn'], 1)  
cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'stn\_hat'] = \  
 m \* cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'n\_mes'] + b  
# #STR  
inicio\_ajuste = '2020-06-01'  
final\_ajuste = '2022-03-01'  
inicio\_proyeccion = '2020-06-01'  
final\_proyeccion = '2028-12-01'  
  
m, b = np.polyfit(cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'n\_mes'],   
cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'str'], 1)  
cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'str\_hat'] = \  
 m \* cargos\_regu.loc[  
 inicio\_proyeccion:final\_proyeccion, 'n\_mes'] + b  
#Restricciones  
inicio\_ajuste = '2020-09-01'  
final\_ajuste = '2022-03-01'  
inicio\_proyeccion = '2020-09-01'  
final\_proyeccion = '2028-12-01'  
  
m, b = np.polyfit(cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'n\_mes'],   
cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'restricciones'], 1)  
cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'restricciones\_hat'] = \  
 m \* cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'n\_mes'] + b  
#Ocv  
inicio\_ajuste = '2021-01-01'  
final\_ajuste = '2022-07-01'  
inicio\_proyeccion = '2021-01-01'  
final\_proyeccion = '2028-12-01'  
  
m, b = np.polyfit(cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'n\_mes'],   
cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'ocv'], 1)  
cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'ocv\_hat'] = \  
 m \* cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'n\_mes'] + b  
#Pérdidas  
inicio\_ajuste = '2022-03-01'  
final\_ajuste = '2022-07-01'  
inicio\_proyeccion = '2022-03-01'  
final\_proyeccion = '2028-12-01'  
  
m, b = np.polyfit(cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'n\_mes'],   
cargos\_regu.loc[inicio\_ajuste:final\_ajuste, 'perdidas'], 1)  
cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'perdidas\_hat'] = \  
 m \* cargos\_regu.loc[inicio\_proyeccion:final\_proyeccion, 'n\_mes'] + b

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.6. Salva y lee cargos\_regu.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# Salva cargos\_regu  
cargos\_regu.to\_parquet\  
 ('parquet/cargos\_regu.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
cargos\_regu.to\_excel('xlsx/cargos\_regu.xlsx', sheet\_name='hoja\_1')  
#Lee cargos\_regu  
cargos\_regu = pd.read\_parquet\  
 ('parquet/cargos\_regu.parquet.gzip', engine='fastparquet')

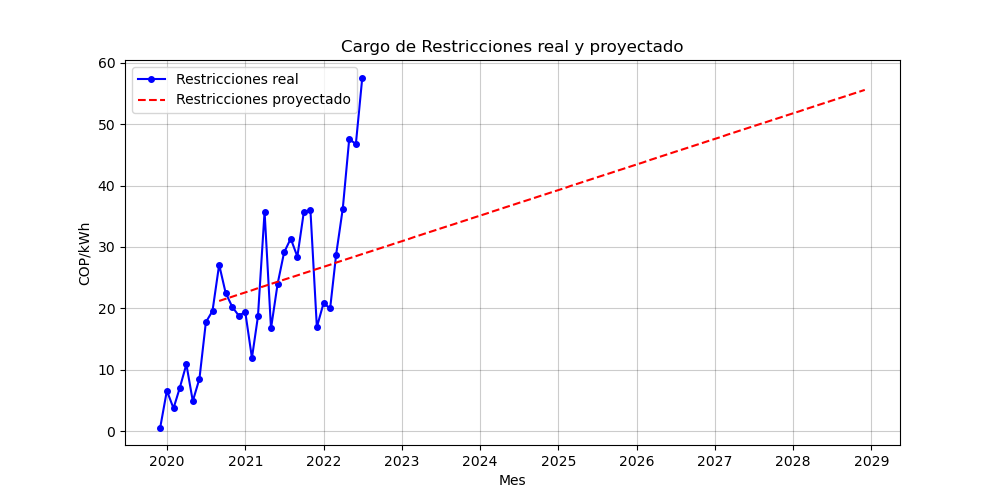
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.7. Grafica STN real y proyectado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Cargo STN real y proyectado')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(cargos\_regu.index, cargos\_regu.stn, 'b-o', \  
 label='STN real', markersize=4)  
ax1.plot(cargos\_regu.index, cargos\_regu.stn\_hat, 'r--', \  
 label='STN proyectado', markersize=2)  
plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2014-01-01'), pd.to\_datetime('2029-01-01'))  
plt.ylim(0, 60)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



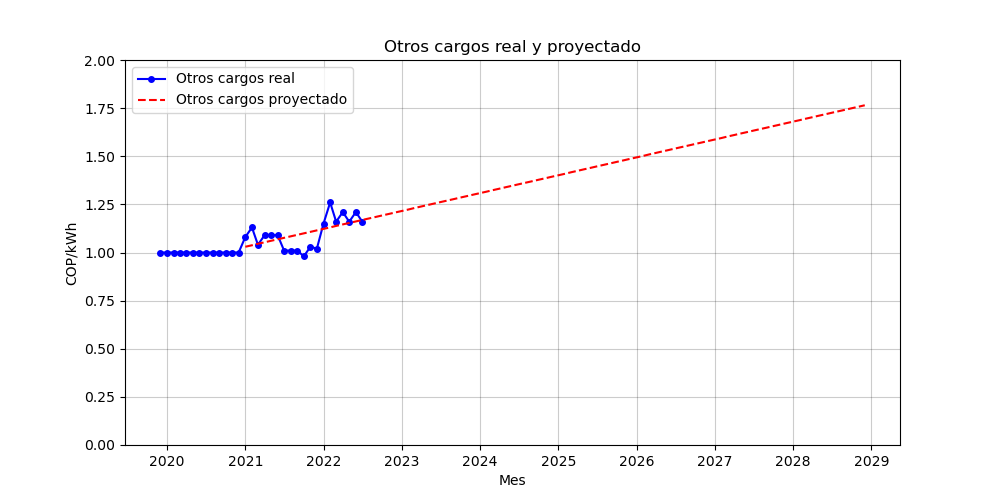
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.8. Grafica STR real y proyectado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Cargo STR real y proyectado')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(cargos\_regu.index, cargos\_regu.str, 'b-o', \  
 label='STR real', markersize=4)  
ax1.plot(cargos\_regu.index, cargos\_regu.str\_hat, 'r--', \  
 label='STR proyectado', markersize=2)  
plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2020-06-01'), pd.to\_datetime('2023-01-01'))  
plt.ylim(0, 60)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



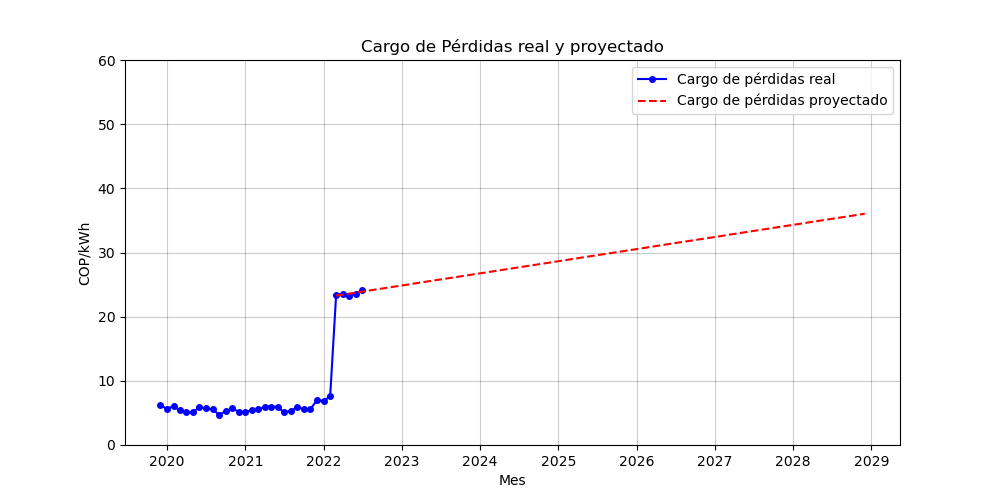
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.9. Grafica del cargo de Restricciones real y proyectado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Cargo de Restricciones real y proyectado')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(cargos\_regu.index, cargos\_regu.restricciones, 'b-o', \  
 label='Restricciones real', markersize=4)  
ax1.plot(cargos\_regu.index, cargos\_regu.restricciones\_hat, 'r--', \  
 label='Restricciones proyectado', markersize=2)  
plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2020-06-01'), pd.to\_datetime('2023-01-01'))  
# plt.ylim(0, 200)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.10. Grafica de Otros cargos, real y proyectado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Otros cargos real y proyectado')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(cargos\_regu.index, cargos\_regu.ocv, 'b-o', \  
 label='Otros cargos real', markersize=4)  
ax1.plot(cargos\_regu.index, cargos\_regu.ocv\_hat, 'r--', \  
 label='Otros cargos proyectado', markersize=2)  
plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2020-01-01'), pd.to\_datetime('2024-01-01'))  
plt.ylim(0, 2)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.11. Grafica del cargo de Pérdidas, real y proyectado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Cargo de Pérdidas real y proyectado')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(cargos\_regu.index, cargos\_regu.perdidas, 'b-o', \  
 label='Cargo de pérdidas real', markersize=4)  
ax1.plot(cargos\_regu.index, cargos\_regu.perdidas\_hat, 'r--', \  
 label='Cargo de pérdidas proyectado', markersize=2)  
plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2022-01-01'), pd.to\_datetime('2023-01-01'))  
plt.ylim(0, 60)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()

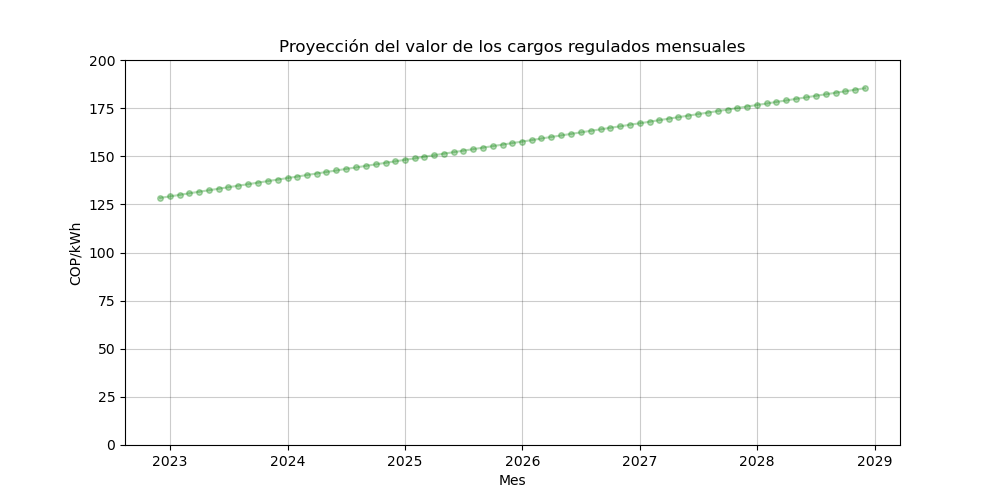


### 6.2.3. Proyecta los egresos de cargos regulados mensuales

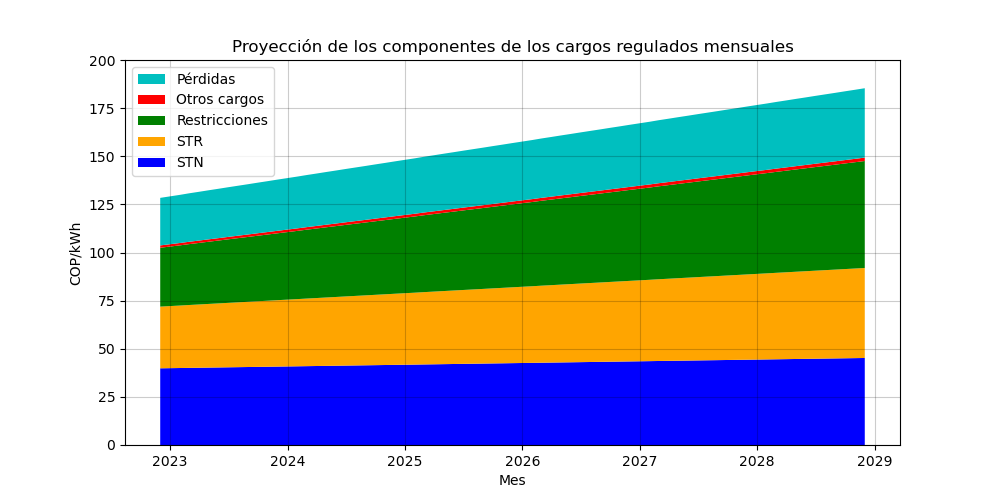
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.12. Crea proyección de egresos por cargos regulados mensuales.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
cargos\_regu = pd.read\_parquet\  
 ('parquet/cargos\_regu.parquet.gzip', engine='fastparquet')  
egresos\_regu\_py = cargos\_regu.loc['2022-12-01':, ['stn\_hat', 'str\_hat',   
 'restricciones\_hat', 'ocv\_hat', 'perdidas\_hat']]  
#Suma de cargos regulados.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='cargos\_regu', value=egresos\_regu\_py.sum(axis=1))  
#Demanda de 11,58 GWh/mes.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='demanda', value=11.58)  
#Egresos de stn.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_stn', value=egresos\_regu\_py.stn\_hat \*   
 egresos\_regu\_py.demanda)   
#Egresos de str.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_str', value=egresos\_regu\_py.str\_hat \*   
 egresos\_regu\_py.demanda)   
#Egresos de restricciones.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_restricciones', value=egresos\_regu\_py.restricciones\_hat \*   
 egresos\_regu\_py.demanda)   
#Egresos de ocv.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_ocv', value=egresos\_regu\_py.ocv\_hat \*   
 egresos\_regu\_py.demanda)   
#Egresos de perdidas.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_perdidas', value=egresos\_regu\_py.perdidas\_hat \*   
 egresos\_regu\_py.demanda)   
#Egresos de cargos regulados.  
egresos\_regu\_py.insert(loc=len(egresos\_regu\_py.columns),   
 column='egresos\_regu', value=egresos\_regu\_py.cargos\_regu \*   
 egresos\_regu\_py.demanda)

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.13. Salva y lee egresos\_regu\_py.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# Salva egresos\_regu\_py  
egresos\_regu\_py.to\_parquet\  
 ('parquet/egresos\_regu\_py.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
egresos\_regu\_py.to\_excel('xlsx/egresos\_regu\_py.xlsx', sheet\_name='hoja\_1')  
#Lee egresos\_regu\_py  
egresos\_regu\_py = pd.read\_parquet\  
 ('parquet/egresos\_regu\_py.parquet.gzip', engine='fastparquet')

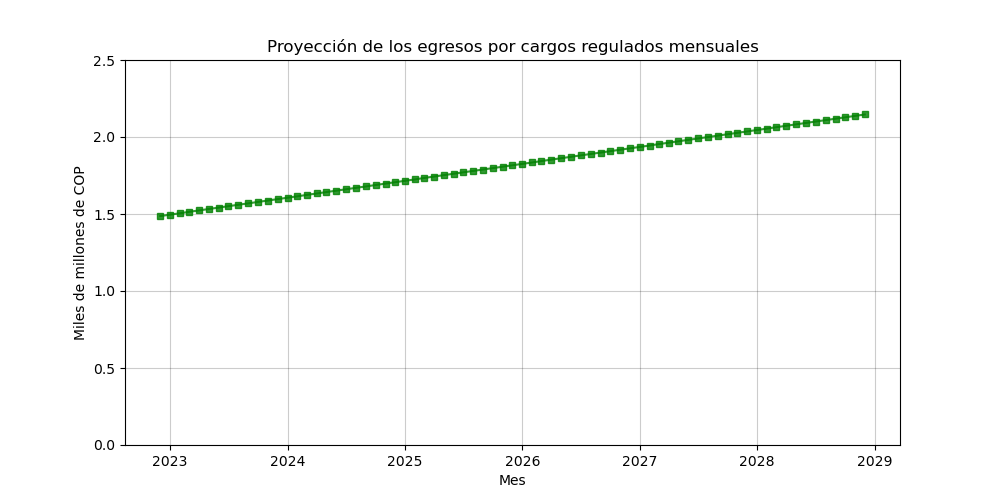
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.14. Grafica la proyección del valor de cargo regulado.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Proyección del valor de los cargos regulados mensuales')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.plot(egresos\_regu\_py.index, egresos\_regu\_py.cargos\_regu, 'g-o', \  
 label='Valor de cargos regulado', markersize=4, alpha=0.3)  
# plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2022-01-01'), pd.to\_datetime('2023-01-01'))  
plt.ylim(0, 200)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.15. Grafica la proyección de los componentes de cargos regulados.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Proyección de los componentes de los cargos regulados mensuales')  
ax1.set\_ylabel ('COP/kWh')  
ax1.set\_xlabel ('Mes')  
ax1.stackplot(egresos\_regu\_py.index, egresos\_regu\_py.stn\_hat,   
 egresos\_regu\_py.str\_hat, egresos\_regu\_py.restricciones\_hat,   
 egresos\_regu\_py.ocv\_hat, egresos\_regu\_py.perdidas\_hat,   
 labels=('STN', 'STR', 'Restricciones', 'Otros cargos', 'Pérdidas'),   
 colors=['b', 'orange', 'g', 'r', 'c'])  
#Cambia el orden de los elementos de la leyenda  
#get handles and labels  
handles, labels = plt.gca().get\_legend\_handles\_labels()  
#specify order of items in legend  
order = [4, 3, 2, 1, 0]  
#add legend to plot  
plt.legend([handles[idx] for idx in order],[labels[idx] for idx in order],   
 loc=2)  
#  
# plt.legend()  
# plt.xlim(pd.to\_datetime('2022-01-01'), pd.to\_datetime('2023-01-01'))  
plt.ylim(0, 200)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.2.16. Grafica la proyección del valor de egresos por cargos regulados.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig = plt.figure(figsize=(10,5))  
ax1 = fig.add\_subplot(1, 1, 1) # Crea una figura conteniendo un solo eje.  
plt.subplots\_adjust(left=None, bottom=None, right=None, top=None, \  
 wspace=None, hspace=None)  
ax1.set\_title ('Proyección de los egresos por cargos regulados mensuales')  
ax1.set\_ylabel ('Miles de millones de COP')  
ax1.set\_xlabel ('Mes')  
ax1.plot(egresos\_regu\_py.index, egresos\_regu\_py.egresos\_regu/1e3, 'g-s', \  
 label='Valor de cargos regulado', markersize=4, alpha=0.8)  
# plt.legend(loc='best')  
# plt.xlim(pd.to\_datetime('2022-01-01'), pd.to\_datetime('2023-01-01'))  
plt.ylim(0, 2.5)  
plt.grid(axis='both', color='k', alpha=0.2)  
plt.show()

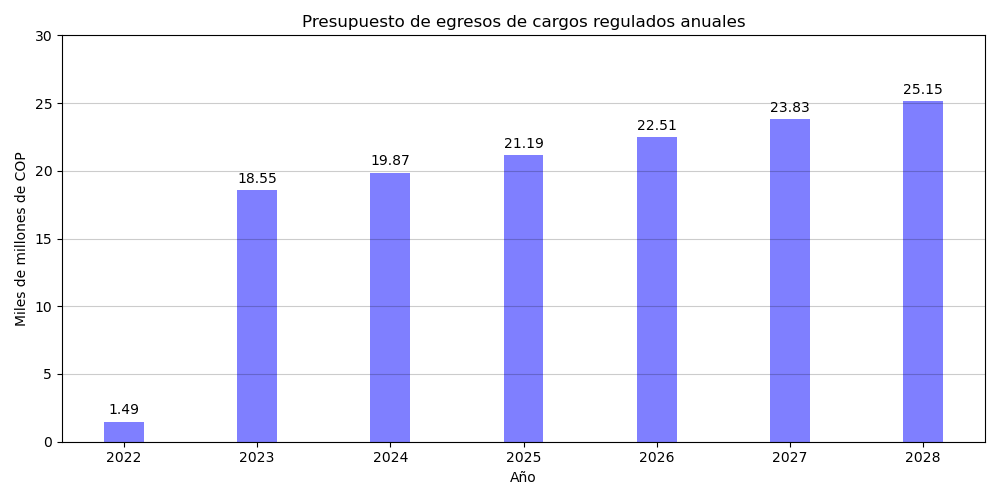


### 6.2.4. Proyecta los egresos de los cargos regulados anuales

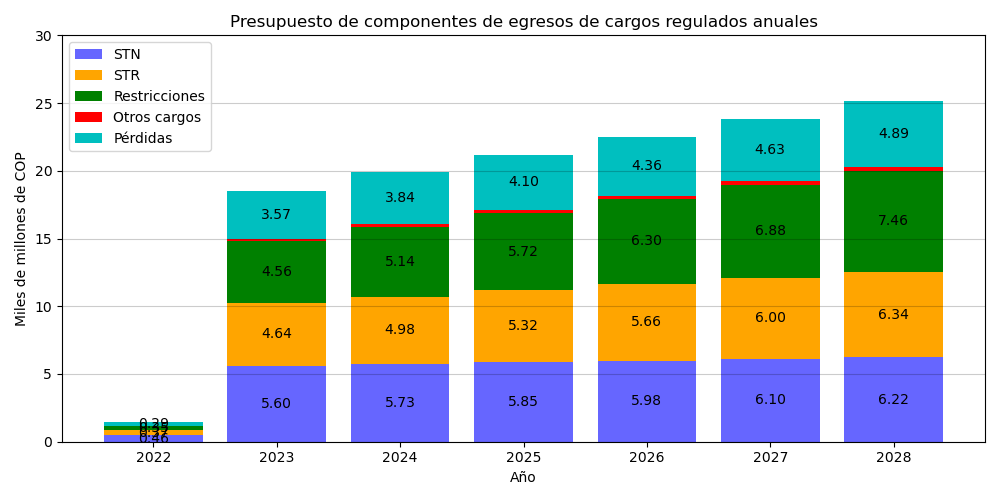
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.17. Calcula los egresos regulados anuales.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egresos\_regu\_py\_a = egresos\_regu\_py.iloc[:, 7:].groupby(  
 egresos\_regu\_py.index.to\_period('Y')).sum()  
egresos\_regu\_py\_a.index = pd.to\_datetime(egresos\_regu\_py\_a.index.astype(str))

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.18. Salva y lee egresos\_regu\_py\_a.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# Salva egresos\_regu\_py\_a  
egresos\_regu\_py\_a.to\_parquet\  
 ('parquet/egresos\_regu\_py\_a.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
egresos\_regu\_py\_a.to\_excel('xlsx/egresos\_regu\_py\_a.xlsx', sheet\_name='hoja\_1')  
#Lee egresos\_regu\_py\_a  
egresos\_regu\_py\_a = pd.read\_parquet\  
 ('parquet/egresos\_regu\_py\_a.parquet.gzip', engine='fastparquet')

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.19. Grafica la proyección de egresos de cargos Regulados anuales.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
anios = egresos\_regu\_py\_a.index.to\_period('Y')  
egresos\_regu\_anual = ((egresos\_regu\_py\_a.egresos\_regu)/1e3).round(2)  
  
  
x = np.arange(len(egresos\_regu\_anual)) # La localización de etiquetas.  
width = 0.30 # El ancho de las barras.  
  
rects1 = ax.bar(x, egresos\_regu\_anual, width,   
label='Egresos regulados anuales', color='b', alpha=0.5)  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto de egresos de cargos regulados anuales')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
ax.set\_xticks(x, anios)  
# ax.legend(loc=2)  
  
ax.bar\_label(rects1, padding=3, fmt='%0.2f', rotation=0)  
  
fig.tight\_layout()  
  
plt.ylim(0, 30)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.20. Grafica los componentes de los egresos de cargos Regulados anuales.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
# egresos\_regu\_anual = ((egresos\_regu\_py\_a.egresos\_regu)/1e3).round(2)  
  
# Crea los datos de las series.  
x = ['2022', '2023', '2024', '2025', '2026', '2027', '2028']  
y1 = ((egresos\_regu\_py\_a.egresos\_stn)/1e3).round(2)  
y2 = ((egresos\_regu\_py\_a.egresos\_str)/1e3).round(2)  
y3 = ((egresos\_regu\_py\_a.egresos\_restricciones)/1e3).round(2)  
y4 = ((egresos\_regu\_py\_a.egresos\_ocv)/1e3).round(2)  
y5 = ((egresos\_regu\_py\_a.egresos\_perdidas)/1e3).round(2)  
   
# Dibuja las barras de manera apilada.  
rectangulo1 = plt.bar(x, y1, color='b', alpha=0.6)  
rectangulo2 = plt.bar(x, y2, bottom=y1, color='orange')  
rectangulo3 = plt.bar(x, y3, bottom=y1+y2, color='g')  
rectangulo4 = plt.bar(x, y4, bottom=y1+y2+y3, color='r')  
rectangulo5 = plt.bar(x, y5, bottom=y1+y2+y3+y4, color='c')  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto de componentes de egresos de cargos regulados anuales')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
# ax.set\_xticks(x, anios)  
# ax.legend(loc=2)  
plt.legend(["STN", "STR", "Restricciones", "Otros cargos", "Pérdidas"], loc=2)  
  
ax.bar\_label(rectangulo1, label\_type='center', fmt='%0.2f', rotation=0)  
ax.bar\_label(rectangulo2, label\_type='center', fmt='%0.2f', rotation=0)  
ax.bar\_label(rectangulo3, label\_type='center', fmt='%0.2f', rotation=0)  
ax.bar\_label(rectangulo5, label\_type='center', fmt='%0.2f', rotation=0)  
  
fig.tight\_layout()  
  
plt.ylim(0, 30)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



## 6.3. Proyección de egresos de la tarifa total con cargos No Regulados más Regulados

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.21. Lee egresos\_regu\_py y egreso\_ENEL\_M2.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
#Lee egresos\_regu\_py  
egresos\_regu\_py = pd.read\_parquet\  
 ('parquet/egresos\_regu\_py.parquet.gzip', engine='fastparquet')  
#Lee egreso\_ENEL\_M2  
egreso\_ENEL\_M2 = pd.read\_parquet\  
 ('parquet/egreso\_ENEL\_M2.parquet.gzip', engine='fastparquet')

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.22. Calcula los egresos totales mensuales.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_total\_m = pd.concat([egreso\_ENEL\_M2.loc[:,   
 ['egreso\_alto', 'egreso\_medio', 'egreso\_bajo']],   
 egresos\_regu\_py.loc[:, ['egresos\_regu']]], axis=1)  
egreso\_total\_m.insert(loc=len(egreso\_total\_m.columns),   
 column='egreso\_total\_alto',   
 value=egreso\_total\_m.egreso\_alto + egreso\_total\_m.egresos\_regu)  
egreso\_total\_m.insert(loc=len(egreso\_total\_m.columns),   
 column='egreso\_total\_medio',   
 value=egreso\_total\_m.egreso\_medio + egreso\_total\_m.egresos\_regu)  
egreso\_total\_m.insert(loc=len(egreso\_total\_m.columns),   
 column='egreso\_total\_bajo',   
 value=egreso\_total\_m.egreso\_bajo + egreso\_total\_m.egresos\_regu)  
egreso\_total\_m.insert(loc=len(egreso\_total\_m.columns),   
 column='egreso\_total\_maximo',   
 value=egreso\_total\_m.loc[:, ['egreso\_total\_alto', 'egreso\_total\_medio',   
 'egreso\_total\_bajo']].max(axis=1))

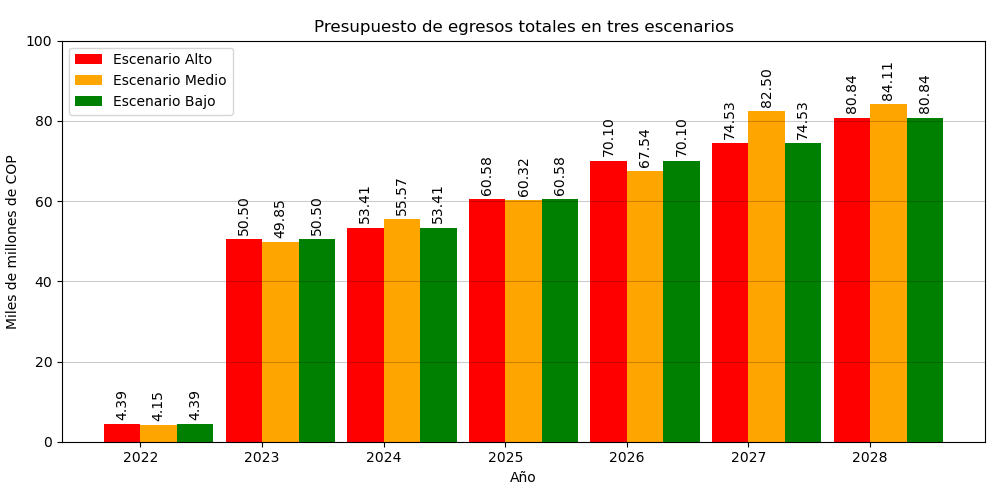
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.23. Calcula el egreso total anual.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_total\_a = egreso\_total\_m.groupby(  
 egreso\_total\_m.index.to\_period('Y')).sum()  
egreso\_total\_a.index = pd.to\_datetime(egreso\_total\_a.index.astype(str))

egreso\_total\_a

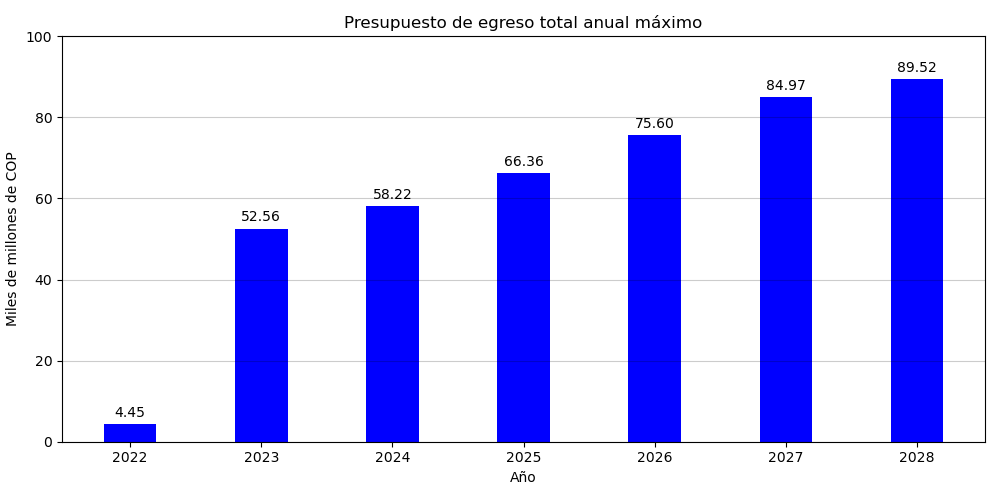
egreso\_alto egreso\_medio egreso\_bajo egresos\_regu \  
index   
2022-01-01 2963.579154 2664.415876 2898.801840 1486.644769   
2023-01-01 33955.052565 31300.276251 31941.698762 18554.239536   
2024-01-01 38254.813013 35700.474030 33534.618618 19873.320713   
2025-01-01 45167.804495 39123.572164 39387.264983 21192.401890   
2026-01-01 52422.563013 45032.874368 47590.172476 22511.483067   
2027-01-01 55325.447199 58674.336434 50694.946532 23830.564244   
2028-01-01 58841.330590 58957.220800 55688.235338 25149.645421   
  
 egreso\_total\_alto egreso\_total\_medio egreso\_total\_bajo \  
index   
2022-01-01 4450.223924 4151.060646 4385.446609   
2023-01-01 52509.292101 49854.515787 50495.938298   
2024-01-01 58128.133726 55573.794743 53407.939331   
2025-01-01 66360.206385 60315.974054 60579.666873   
2026-01-01 74934.046081 67544.357436 70101.655543   
2027-01-01 79156.011443 82504.900678 74525.510776   
2028-01-01 83990.976012 84106.866221 80837.880759   
  
 egreso\_total\_maximo   
index   
2022-01-01 4450.223924   
2023-01-01 52562.277612   
2024-01-01 58218.944591   
2025-01-01 66360.206385   
2026-01-01 75595.843039   
2027-01-01 84966.485613   
2028-01-01 89519.005706

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
# 6.2.24. Salva y lee egreso\_total\_m y egreso\_total\_a.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# Salva egresos\_total\_m  
egreso\_total\_m.to\_parquet\  
 ('parquet/egreso\_total\_m.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
egreso\_total\_m.to\_excel('xlsx/egreso\_total\_m.xlsx', sheet\_name='hoja\_1')  
# Salva egresos\_total\_a  
egreso\_total\_a.to\_parquet\  
 ('parquet/egreso\_total\_a.parquet.gzip', \  
 compression='gzip', engine='fastparquet')  
egreso\_total\_a.to\_excel('xlsx/egreso\_total\_a.xlsx', sheet\_name='hoja\_1')  
##  
#Lee egreso\_total\_m  
egreso\_total\_m = pd.read\_parquet\  
 ('parquet/egreso\_total\_m.parquet.gzip', engine='fastparquet')  
#Lee egreso\_total\_a  
egreso\_total\_a = pd.read\_parquet\  
 ('parquet/egreso\_total\_a.parquet.gzip', engine='fastparquet')

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.25. Grafica el egreso total anual en tres escenarios.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
anios = egreso\_total\_a.index.to\_period('Y')  
escenario\_alto = ((egreso\_total\_a.egreso\_total\_bajo)/1e3).round(2)  
escenario\_medio = ((egreso\_total\_a.egreso\_total\_medio)/1e3).round(2)  
escenario\_bajo = ((egreso\_total\_a.egreso\_total\_bajo)/1e3).round(2)  
  
  
x = np.arange(len(escenario\_alto)) # La localización de etiquetas.  
width = 0.30 # El ancho de las barras.  
  
rects1 = ax.bar(x - width/2, escenario\_alto, width, label='Escenario Alto', \  
 color='r')  
rects2 = ax.bar(x + width/2, escenario\_medio, width, label='Escenario Medio', \  
 color='orange')  
rects3 = ax.bar(x + 3 \* width/2, escenario\_bajo, width, label='Escenario Bajo', \  
 color='g')  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto de egresos totales en tres escenarios')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
ax.set\_xticks(x, anios)  
ax.legend(loc=2)  
  
ax.bar\_label(rects1, padding=3, fmt='%0.2f', rotation=90)  
ax.bar\_label(rects2, padding=3, fmt='%0.2f', rotation=90)  
ax.bar\_label(rects3, padding=3, fmt='%0.2f', rotation=90)  
  
fig.tight\_layout()  
  
plt.ylim(0, 100)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.26. Grafica el máximo egreso anual por cargos No Regulados.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
fig, ax = plt.subplots(figsize=(10,5))  
  
anios = egreso\_total\_a.index.to\_period('Y')  
egreso\_total\_maximo = ((egreso\_total\_a.egreso\_total\_maximo)/1e3).round(2)  
  
  
x = np.arange(len(egreso\_total\_maximo)) # La localización de etiquetas.  
width = 0.40 # El ancho de las barras.  
  
rects1 = ax.bar(x, egreso\_total\_maximo, width,   
 label='Egreso total máximo', color='b')  
  
# Add some text for labels, title and custom x-axis tick labels, etc.  
ax.set\_title('Presupuesto de egreso total anual máximo')  
ax.set\_xlabel('Año')  
ax.set\_ylabel('Miles de millones de COP')  
ax.set\_xticks(x, anios)  
# ax.legend(loc=2)  
  
ax.bar\_label(rects1, padding=3, fmt='%0.2f', rotation=0)  
  
fig.tight\_layout()  
  
plt.ylim(0, 100)  
plt.grid(axis='y', color='k', alpha=0.2)  
plt.show()



#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
# 6.1.27. Calcula el valor estimado del contrato de suministro de energía.  
#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
egreso\_total\_a.egreso\_total\_maximo.sum(axis=0)

431672.9868704862

El valor estimado del contrato durante el periodo comprendio entre diciembre de 2022 y diciembre de 2028 es 431.672 millones de pesos.