In [32]:
```
!pip install PyPortfolioOpt
!pip install portfolio-backtest
!pip install riskfolio-lib
!pip install yesg
```

```
Requirement already satisfied: PyPortfolioOpt in /usr/local/lib/python3.1
0/dist-packages (1.5.6)
Requirement already satisfied: cvxpy>=1.1.19 in /usr/local/lib/python3.10/
dist-packages (from PyPortfolioOpt) (1.6.0)
Requirement already satisfied: ecos<3.0.0,>=2.0.14 in /usr/local/lib/pytho
n3.10/dist-packages (from PyPortfolioOpt) (2.0.14)
Requirement already satisfied: numpy>=1.26.0 in /usr/local/lib/python3.10/
dist-packages (from PyPortfolioOpt) (1.26.4)
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.10/d
ist-packages (from PyPortfolioOpt) (2.2.2)
Requirement already satisfied: plotly<6.0.0,>=5.0.0 in /usr/local/lib/pyth
on3.10/dist-packages (from PyPortfolioOpt) (5.24.1)
Requirement already satisfied: scipy>=1.3 in /usr/local/lib/python3.10/dis
t-packages (from PyPortfolioOpt) (1.13.1)
Requirement already satisfied: osqp>=0.6.2 in /usr/local/lib/python3.10/di
st-packages (from cvxpy>=1.1.19->PyPortfolioOpt) (0.6.7.post3)
Requirement already satisfied: clarabel>=0.5.0 in /usr/local/lib/python3.1
0/dist-packages (from cvxpy>=1.1.19->PyPortfolioOpt) (0.9.0)
Requirement already satisfied: scs>=3.2.4.post1 in /usr/local/lib/python3.
10/dist-packages (from cvxpy>=1.1.19->PyPortfolioOpt) (3.2.7)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/py
thon3.10/dist-packages (from pandas>=0.19->PyPortfolioOpt) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/d
ist-packages (from pandas>=0.19->PyPortfolioOpt) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.1
0/dist-packages (from pandas>=0.19->PyPortfolioOpt) (2024.2)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.1
0/dist-packages (from plotly<6.0.0,>=5.0.0->PyPortfolioOpt) (9.0.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist
-packages (from plotly<6.0.0,>=5.0.0->PyPortfolioOpt) (24.2)
Requirement already satisfied: qdldl in /usr/local/lib/python3.10/dist-pac
kages (from osqp>=0.6.2->cvxpy>=1.1.19->PyPortfolioOpt) (0.1.7.post4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.2->pandas>=0.19->PyPortfolioOpt) (1.1
7.0)
Requirement already satisfied: portfolio-backtest in /usr/local/lib/python
3.10/dist-packages (0.3.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-pa
ckages (from portfolio-backtest) (2.2.2)
Requirement already satisfied: yfinance in /usr/local/lib/python3.10/dist-
packages (from portfolio-backtest) (0.2.50)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dis
t-packages (from portfolio-backtest) (3.8.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/d
ist-packages (from portfolio-backtest) (1.6.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib->portfolio-backtest) (1.3.1)
```

```
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/d
ist-packages (from matplotlib->portfolio-backtest) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python
3.10/dist-packages (from matplotlib->portfolio-backtest) (4.55.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python
3.10/dist-packages (from matplotlib->portfolio-backtest) (1.4.7)
Requirement already satisfied: numpy<2,>=1.21 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib->portfolio-backtest) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib->portfolio-backtest) (24.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/
dist-packages (from matplotlib->portfolio-backtest) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib->portfolio-backtest) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/pyth
on3.10/dist-packages (from matplotlib->portfolio-backtest) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/d
ist-packages (from pandas->portfolio-backtest) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.1
0/dist-packages (from pandas->portfolio-backtest) (2024.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/d
ist-packages (from scikit-learn->portfolio-backtest) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/
dist-packages (from scikit-learn->portfolio-backtest) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/pyth
on3.10/dist-packages (from scikit-learn->portfolio-backtest) (3.5.0)
Requirement already satisfied: requests>=2.31 in /usr/local/lib/python3.1
0/dist-packages (from yfinance->portfolio-backtest) (2.32.3)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/pytho
n3.10/dist-packages (from yfinance->portfolio-backtest) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in /usr/local/lib/python3.10/di
st-packages (from yfinance->portfolio-backtest) (5.3.0)
Requirement already satisfied: platformdirs>=2.0.0 in /usr/local/lib/pytho
n3.10/dist-packages (from yfinance->portfolio-backtest) (4.3.6)
Requirement already satisfied: frozendict>=2.3.4 in /usr/local/lib/python
3.10/dist-packages (from yfinance->portfolio-backtest) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in /usr/local/lib/python3.1
0/dist-packages (from yfinance->portfolio-backtest) (3.17.8)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /usr/local/lib/py
thon3.10/dist-packages (from yfinance->portfolio-backtest) (4.12.3)
Requirement already satisfied: html5lib>=1.1 in /usr/local/lib/python3.10/
dist-packages (from yfinance->portfolio-backtest) (1.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/
dist-packages (from beautifulsoup4>=4.11.1->yfinance->portfolio-backtest)
(2.6)
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.10/dist-
packages (from html5lib>=1.1->yfinance->portfolio-backtest) (1.17.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/d
ist-packages (from html5lib>=1.1->yfinance->portfolio-backtest) (0.5.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/
python3.10/dist-packages (from requests>=2.31->yfinance->portfolio-backtes
t) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/d
```

```
ist-packages (from requests>=2.31->yfinance->portfolio-backtest) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python
3.10/dist-packages (from requests>=2.31->yfinance->portfolio-backtest) (2.
2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python
3.10/dist-packages (from requests>=2.31->yfinance->portfolio-backtest) (20
24.12.14)
Requirement already satisfied: riskfolio-lib in /usr/local/lib/python3.10/
dist-packages (6.3.1)
Requirement already satisfied: numpy>=1.24.0 in /usr/local/lib/python3.10/
dist-packages (from riskfolio-lib) (1.26.4)
Requirement already satisfied: scipy>=1.10.0 in /usr/local/lib/python3.10/
dist-packages (from riskfolio-lib) (1.13.1)
Requirement already satisfied: pandas>=2.0.0 in /usr/local/lib/python3.10/
dist-packages (from riskfolio-lib) (2.2.2)
Requirement already satisfied: matplotlib>=3.8.0 in /usr/local/lib/python
3.10/dist-packages (from riskfolio-lib) (3.8.0)
Requirement already satisfied: clarabel>=0.6.0 in /usr/local/lib/python3.1
0/dist-packages (from riskfolio-lib) (0.9.0)
Requirement already satisfied: cvxpy>=1.5.2 in /usr/local/lib/python3.10/d
ist-packages (from riskfolio-lib) (1.6.0)
Requirement already satisfied: scikit-learn>=1.3.0 in /usr/local/lib/pytho
n3.10/dist-packages (from riskfolio-lib) (1.6.0)
Requirement already satisfied: statsmodels>=0.13.5 in /usr/local/lib/pytho
n3.10/dist-packages (from riskfolio-lib) (0.14.4)
Requirement already satisfied: arch>=7.0 in /usr/local/lib/python3.10/dist
-packages (from riskfolio-lib) (7.2.0)
Requirement already satisfied: xlsxwriter>=3.1.2 in /usr/local/lib/python
3.10/dist-packages (from riskfolio-lib) (3.2.0)
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.10/
dist-packages (from riskfolio-lib) (3.4.2)
Requirement already satisfied: astropy>=5.1 in /usr/local/lib/python3.10/d
ist-packages (from riskfolio-lib) (6.1.7)
Requirement already satisfied: pybind11>=2.10.1 in /usr/local/lib/python3.
10/dist-packages (from riskfolio-lib) (2.13.6)
Requirement already satisfied: pyerfa>=2.0.1.1 in /usr/local/lib/python3.1
0/dist-packages (from astropy>=5.1->riskfolio-lib) (2.0.1.5)
Requirement already satisfied: astropy-iers-data>=0.2024.10.28.0.34.7 in /
usr/local/lib/python3.10/dist-packages (from astropy>=5.1->riskfolio-lib)
(0.2024.12.16.0.35.48)
Requirement already satisfied: PyYAML>=3.13 in /usr/local/lib/python3.10/d
ist-packages (from astropy>=5.1->riskfolio-lib) (6.0.2)
Requirement already satisfied: packaging>=19.0 in /usr/local/lib/python3.1
0/dist-packages (from astropy>=5.1->riskfolio-lib) (24.2)
Requirement already satisfied: osqp>=0.6.2 in /usr/local/lib/python3.10/di
st-packages (from cvxpy>=1.5.2->riskfolio-lib) (0.6.7.post3)
Requirement already satisfied: scs>=3.2.4.post1 in /usr/local/lib/python3.
10/dist-packages (from cvxpy>=1.5.2->riskfolio-lib) (3.2.7)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/d
ist-packages (from matplotlib>=3.8.0->riskfolio-lib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python
```

```
3.10/dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python
3.10/dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (1.4.7)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/
dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.
10/dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/pyth
on3.10/dist-packages (from matplotlib>=3.8.0->riskfolio-lib) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/d
ist-packages (from pandas>=2.0.0->riskfolio-lib) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.1
0/dist-packages (from pandas>=2.0.0->riskfolio-lib) (2024.2)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/
dist-packages (from scikit-learn>=1.3.0->riskfolio-lib) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/pyth
on3.10/dist-packages (from scikit-learn>=1.3.0->riskfolio-lib) (3.5.0)
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/d
ist-packages (from statsmodels>=0.13.5->riskfolio-lib) (1.0.1)
Requirement already satisfied: qdldl in /usr/local/lib/python3.10/dist-pac
kages (from osqp>=0.6.2->cvxpy>=1.5.2->riskfolio-lib) (0.1.7.post4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib>=3.8.0->riskfolio-lib) (1.
17.0)
Requirement already satisfied: yesg in /usr/local/lib/python3.10/dist-pack
ages (2.1.1)
```

Now that our backtest has been carried out and confirmed the stability of our portfolio's performance over time, we will now show how, once we have obtained the weights of our risk parity portfolio on the ETFs, we can extract the weights of the stocks that make them up. We illustrate this with the weights of the ERC trained over the period January to September 2024 and we will calculate the weights that we will hold over the October, November and December quarters.

In [ ]:
```
from GeneralFunction import download_data,calculate_metrics,split_data,we
import yfinance as yf
import certifi
import requests
from io import BytesIO, StringIO
import pandas as pd
import yesg
import warnings
warnings.filterwarnings("ignore")
```

In [36]:
```
params = {
        "i": 63 ,
        "j": 189,
        "k": 63,
        "start_date": "2024-01-01",
        "end_date": "2024-06-06",
        "tickers": ['SPHQ', 'IVE', 'SPYD', 'SPLV', 'SPMO']
```

```
        }


returns = download_data(params["tickers"], params["start_date"], params["
weights = optimize_risk_par(returns)
```

```
[**********************100%************************]  5 of 5 completed
```

In [37]:
```
def download_data(url):
    """
    Télécharge et retourne les données d'une URL, en gérant les fichiers
    """
    if url.endswith('.xls') or url.endswith('.xlsx'):
        skip_rows = 4
        response = requests.get(url)
        response.raise_for_status()
        data = pd.read_excel(BytesIO(response.content), skiprows=skip_row
    else:
        skip_rows = 0
        response = requests.get(url)
        response.raise_for_status()
        csv_data = response.content.decode('utf-8')
        try:
            data = pd.read_csv(StringIO(csv_data), skiprows=skip_rows)
        except pd.errors.ParserError:
            try:
                data = pd.read_csv(StringIO(csv_data), delimiter=';', ski
            except pd.errors.ParserError:
                data = pd.read_csv(StringIO(csv_data), skiprows=9 + skip_

    if not any('weight' in col.lower() for col in data.columns):
        data = pd.read_csv(StringIO(csv_data), skiprows=9 + skip_rows)
    return data

def normalize_ticker(ticker):
    """
    Normalise le ticker en supprimant les espaces et les caractères inuti
    """
    if pd.isna(ticker):  # Vérifier si ticker est NaN
        return ''
    return ticker.replace('/', '').strip().upper()

def calculate_total_weights_per_etf(url, etf_weight):
    """
    Calcule le poids total pour chaque action d'un fichier CSV ou Excel.
    """
    data = download_data(url)
    weight_column = next((col for col in data.columns if 'weight' in col.
    if weight_column is None:
        print("Colonne 'Weight' ou 'Weight (%)' introuvable dans le fichi
        return None

    holding_ticker_column = next((col for col in data.columns if 'holding
    ticker_column = next((col for col in data.columns if 'ticker' in col.
```

```
    if holding_ticker_column:
        ticker_column = holding_ticker_column
    elif not ticker_column:
        print("Aucune colonne 'Ticker' ou 'Holding Ticker' introuvable da
        return None

    data['Weighted_Weight'] = data[weight_column] * etf_weight

    if 'Name' in data.columns:
        data = data[~data['Name'].isin(["US DOLLAR", "Cash/Receivables/Pa

    # Normaliser les tickers
    data[ticker_column] = data[ticker_column].apply(normalize_ticker)

    total_weights = data.groupby(ticker_column)[['Name', 'Weighted_Weight
    total_weights.columns = ['Ticker', 'Name', 'Total_Weight']
    total_weights = total_weights[total_weights['Total_Weight'] > 0]
    return total_weights

def calculate_total_weights(weights):
    # Téléchargement et traitement des données pour chaque URL
    urls = [
        "https://www.invesco.com/us/financial-products/etfs/holdings/main
        "https://www.invesco.com/us/financial-products/etfs/holdings/main
        "https://www.invesco.com/us/financial-products/etfs/holdings/main
        "https://www.ishares.com/us/products/239728/ishares-sp-500-value-
        "https://www.ssga.com/us/en/intermediary/library-content/products
    ]

    final_data = pd.DataFrame()

    # Appliquer chaque URL et poids d'ETF
    for url, weight in zip(urls, weights):
        etf_data = calculate_total_weights_per_etf(url, weight)
        if etf_data is not None:
            final_data = pd.concat([final_data, etf_data])

    # Étape finale : Regrouper par ticker exact et sommer les poids
    final_data = final_data.groupby('Ticker', as_index=False)['Total_Weig
    final_data['Total_Weight'] /= final_data['Total_Weight'].sum()

    # Trier les données en ordre décroissant de poids
    #final_data.sort_values(by='Total_Weight', ascending=False, inplace=T

    # Vérification des doublons
    duplicate_tickers = final_data[final_data.duplicated(subset='Ticker',
    if not duplicate_tickers.empty:
        print("Doublons trouvés dans les tickers :")
        print(duplicate_tickers)
    #else:
    #    print("Aucun doublon trouvé dans les tickers.")

    # Affichage du résultat final
```

```
    #print(final_data.to_string(index=False))

    # Enregistrer dans un fichier CSV
    #final_data.to_csv("poids_actions_etf.csv", index=False)
    #print("Les données complètes ont été enregistrées dans 'poids_action

    final_data.columns = ['Symbol', 'Weights']
    final_data = final_data.set_index('Symbol')

    return final_data

#Stocks with several classes: fox and news corp
```

In [38]:
```
import yfinance as yf
import pandas as pd

def calculate_portfolio_value(weights: pd.DataFrame):
    """
    Calculate the weighted value of stocks in a portfolio using Yahoo Fin

    Args:
        weights (pd.DataFrame): A DataFrame indexed by stock symbols, wit

    Returns:
        pd.DataFrame: A DataFrame with columns ['Symbol', 'Weight', 'Last
    """
    # Ensure the DataFrame has the expected structure
    if 'Weights' not in weights.columns:
        raise ValueError("The input DataFrame must have a 'Weights' colum

    results = []

    for symbol, row in weights.iterrows():
        weight = row['Weights']

        # Fetch the stock data from Yahoo Finance
        try:
            stock = yf.Ticker(symbol)
            last_close = stock.history(period='1d')['Close'].iloc[-1]

            # Calculate the weighted value
            weighted_value = weight * last_close
            results.append({'Symbol': symbol, 'Weight': weight, 'Last Clo
        except Exception as e:
            print(f"Error fetching data for {symbol}: {e}")

    # Return results as a DataFrame
    return pd.DataFrame(results)
```

In [39]:
```
stock_weights = calculate_total_weights(weights)
stock_weights.sort_values(by='Weights', ascending=False)
```

Out[39]:

Symbol	Weights
NVDA	0.030750
AMZN	0.029636
AAPL	0.027115
META	0.023787
BRKB	0.018218
...	...
PNR	0.000037
ROL	0.000037
GNRC	0.000019
ADSK	0.000004
PANW	0.000004

469 rows × 1 columns

In [40]:
```
calculate_portfolio_value(stock_weights)
```

```
ERROR:yfinance:$BFB: possibly delisted; no price data found  (period=1d) (
Yahoo error = "No data found, symbol may be delisted")
Error fetching data for BFB: single positional indexer is out-of-bounds
ERROR:yfinance:$BRKB: possibly delisted; no price data found  (period=1d)
(Yahoo error = "No data found, symbol may be delisted")
Error fetching data for BRKB: single positional indexer is out-of-bounds
ERROR:yfinance:Could not get exchangeTimezoneName for ticker 'CTAS' reaso
n: 'chart'
ERROR:yfinance:$CTAS: possibly delisted; no price data found  (period=1d)
Error fetching data for CTAS: single positional indexer is out-of-bounds
ERROR:yfinance:$XTSLA: possibly delisted; no price data found  (period=1d)
(Yahoo error = "No data found, symbol may be delisted")
Error fetching data for XTSLA: single positional indexer is out-of-bounds
```

Out[40]:

	Symbol	Weight	Last Close	Weighted Value
0	A	0.001090	134.509995	0.146642
1	AAPL	0.027115	254.490005	6.900543
2	ABBV	0.003709	175.580002	0.651290
3	ABT	0.003497	114.230003	0.399469
4	ACGL	0.000793	90.989998	0.072169
...	...	...	...	...
460	XYL	0.000224	117.139999	0.026278
461	YUM	0.002353	132.360001	0.311505
462	ZBH	0.000168	107.120003	0.018023
463	ZBRA	0.000093	393.040009	0.036738
464	ZTS	0.002000	164.839996	0.329708

465 rows × 4 columns

Constraints

- We do not take into account the worst stocks in terms of ESG scores
- Sector constraints (the s&p is already concentrated so it might be interesting to allow a difference with the sector wiehgts in the original s&p 500)
- Concentration limits (a limit for each stock)
- Liquidity constraints (use only stocks with volume higher than a threshold): not so interesting because all s&p 500 stocks are liquid no short allowed
- tracking error: determined with backtest

In [41]:
```
def esg_constraints(stock_weights, quantile_threshold):
    """
    quantile_threshold: float between 0 and 1 to keep the best quantile_t
    stock_weights: DataFrame with the weights of the stocks in the portfo
    """
    # get the s&p 500 tickers with their name, sector and sub-industry
    df_sp500 = pd.read_html('https://en.wikipedia.org/wiki/List_of_S%26P_
    df_sp500 = df_sp500.set_index('Symbol') # set the index to be the sym

    df_sp500.drop(index='GOOG', inplace=True) # drop the GOOG row because

    df_esg = df_sp500.copy()
    for ticker in df_sp500.index:
        try:
            df_esg.loc[ticker, "ESG Score"] = yesg.get_historic_esg(ticke
        except AttributeError:
```

```
            pass

    #drop all the stocks that have no ESG score
    df_esg.dropna(axis=0, inplace=True)

    #The best ESG score is 0
    #drop the worst stocks in terms of ESG score
    df_esg = df_esg[df_esg['ESG Score'] < df_esg['ESG Score'].quantile(qu

    #drop all the stocks that have an ESG score above the threshold
    #threshold = 30
    #df_esg = df_esg[df_esg['ESG Score'] < threshold]

    df_esg = df_esg.merge(stock_weights, left_index=True, right_index=Tru

    df_esg['Weights'] /= df_esg['Weights'].sum()

    return df_esg
```

In [42]:
```
quantile_threshold = 0.9
df_esg = esg_constraints(stock_weights, quantile_threshold)
df_esg.sort_values(by='Weights', ascending=False)
```

```
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
An error has occurred. The ticker symbol might be wrong or you might need
to wait to continue.
```

Out[42]:

Symbol	Security	GICS Sector	GICS Sub-Industry	ESG Score	Weights
NVDA	Nvidia	Information Technology	Semiconductors	12.23	0.034653
AMZN	Amazon	Consumer Discretionary	Broadline Retail	29.01	0.033398
AAPL	Apple Inc.	Information Technology	Technology Hardware, Storage & Peripherals	16.79	0.030557
COST	Costco	Consumer Staples	Consumer Staples Merchandise Retail	29.12	0.019084
AVGO	Broadcom	Information Technology	Semiconductors	19.20	0.017354
...	...	...	...	...	...
ROL	Rollins, Inc.	Industrials	Environmental & Facilities Services	18.60	0.000042
ALLE	Allegion	Industrials	Building Products	19.83	0.000042
GNRC	Generac	Industrials	Electrical Components & Equipment	21.93	0.000021
PANW	Palo Alto Networks	Information Technology	Systems Software	13.56	0.000005
ADSK	Autodesk	Information Technology	Application Software	15.14	0.000005

412 rows × 5 columns

Sector constraints

In [43]:

```
def sector_constraints(df, sector_threshold):
    """
    df: DataFrame following the same format as the one returned by esg_co
    sector_threshold: float between 0 and 1 that a sector cannot exceed i
    """
    #We calculate the weights of the stocks in each sector
    df_weights_by_sector = df[['GICS Sector', 'Weights']].groupby('GICS S

    #We create a list of the sectors that have a weight above the thresho
    sectors_above_threshold = df_weights_by_sector.loc[df_weights_by_sect
    while len(sectors_above_threshold) > 0:
        for sector in sectors_above_threshold:
            #for each stock in the sector, we apply a factor such that th
            df_sector = df[df['GICS Sector'] == sector]
```

```
            factor = sector_threshold / df_weights_by_sector.loc[sector,
            df.loc[df_sector.index, 'Weights'] = df_sector['Weights'] * f

        #we increase the weights of the stocks in the other sectors to ke
        df.loc[~df['GICS Sector'].isin(sectors_above_threshold), "Weights

        #Some sectors may now have a weight above the threshold so we rep
        df_weights_by_sector = df[['GICS Sector', 'Weights']].groupby('GI
        sectors_above_threshold = df_weights_by_sector.loc[df_weights_by_

    return df
```

In [44]:
```
sector_threshold = 0.17
weights_after_sector = sector_constraints(df_esg, sector_threshold)

#Check that none of the sectors have a weight above the threshold
#weights_after_sector[['GICS Sector', 'Weights']].groupby('GICS Sector').

#Check that the sum of the weights is equal to 1
#weights_after_sector[['GICS Sector', 'Weights']].groupby('GICS Sector').

weights_after_sector.sort_values(by='Weights', ascending=False)
```

Out[44]:

Symbol	Security	GICS Sector	GICS Sub-Industry	ESG Score	Weights
AMZN	Amazon	Consumer Discretionary	Broadline Retail	29.01	0.034318
NVDA	Nvidia	Information Technology	Semiconductors	12.23	0.033441
AAPL	Apple Inc.	Information Technology	Technology Hardware, Storage & Peripherals	16.79	0.029488
COST	Costco	Consumer Staples	Consumer Staples Merchandise Retail	29.12	0.019609
JNJ	Johnson & Johnson	Health Care	Pharmaceuticals	20.10	0.017793
...	...	...	...	...	...
INCY	Incyte	Health Care	Biotechnology	23.71	0.000043
PNR	Pentair	Industrials	Industrial Machinery & Supplies & Components	22.00	0.000043
GNRC	Generac	Industrials	Electrical Components & Equipment	21.93	0.000022
PANW	Palo Alto Networks	Information Technology	Systems Software	13.56	0.000005
ADSK	Autodesk	Information Technology	Application Software	15.14	0.000005

412 rows × 5 columns

Stock constraints

In [45]:
```
def stock_constraints(df, stock_threshold):
    """
    df: DataFrame following the same format as the one returned by sector
    stock_threshold: float between 0 and 1 that a stock cannot exceed in
    """
    #We create a list of the stocks that have a weight above the threshol
    stocks_above_threshold = df.loc[df['Weights'] > stock_threshold].inde
    while len(stocks_above_threshold) > 0:
        #we set the weights of the stocks above the threshold to the thre
        df.loc[stocks_above_threshold, 'Weights'] = stock_threshold
        #we increase (proportionally) the weights of the other stocks to
        df.loc[~df.index.isin(stocks_above_threshold), "Weights"] /= df.l
        #Some stocks may now have a weight above the threshold so we repe
        stocks_above_threshold = df.loc[df['Weights'] > stock_threshold].
```

```
    return df
```

In [46]:
```
#print("Max weight for a stock before stock constraints: ", weights_after
stock_threshold = 0.05
weights_after_stock = stock_constraints(weights_after_sector, stock_thres
#print("Max weight for a stock after stock constraints: ", weights_after_

#Check that none of the stocks have a weight above the threshold
#weights_after_stock['Weights'].max()

#Check that the sum of the weights is equal to 1
#weights_after_stock['Weights'].sum()

weights_after_stock.sort_values(by='Weights', ascending=False)
```

Out[46]:

Symbol	Security	GICS Sector	GICS Sub-Industry	ESG Score	Weights
AMZN	Amazon	Consumer Discretionary	Broadline Retail	29.01	0.034318
NVDA	Nvidia	Information Technology	Semiconductors	12.23	0.033441
AAPL	Apple Inc.	Information Technology	Technology Hardware, Storage & Peripherals	16.79	0.029488
COST	Costco	Consumer Staples	Consumer Staples Merchandise Retail	29.12	0.019609
JNJ	Johnson & Johnson	Health Care	Pharmaceuticals	20.10	0.017793
...	...	...	...	...	...
INCY	Incyte	Health Care	Biotechnology	23.71	0.000043
PNR	Pentair	Industrials	Industrial Machinery & Supplies & Components	22.00	0.000043
GNRC	Generac	Industrials	Electrical Components & Equipment	21.93	0.000022
PANW	Palo Alto Networks	Information Technology	Systems Software	13.56	0.000005
ADSK	Autodesk	Information Technology	Application Software	15.14	0.000005

412 rows × 5 columns

References

Implementations of equal risk contribution

- https://github.com/matthewgilbert/erc/blob/master/erc/erc.py
- https://github.com/mirca/riskparity.py (not used)
- https://thequantmba.wordpress.com/2016/12/14/risk-parityrisk-budgeting-portfolio-in-python/

Papers

- Paper of Maillard, Roncalli and Teiletche
- Slides of Maillard, Roncalli and Teiletche
- Master's thesis of David Stefanovits