

Elspot Supply and Demand

August 23, 2020

1 Supply and Demand for Elspot

NordPool has provided the following information on the Elspot System Price bid curves. The document below describes how to adjust the buy and sell volumes using accepted blocks buy/sell and net flows.

```
[1]: from IPython.display import IFrame
fileURL = "https://www.nordpoolgroup.com/globalassets/
↳information-in-market-cross-point-data-reports.pdf "
IFrame(fileURL, width=900, height=300)
```

```
[1]: <IPython.lib.display.IFrame at 0x10f9efb50>
```

1.1 Cleaning data

```
[2]: import numpy as np
import pandas as pd
```

```
[3]: def cleaningData(date, hour):

    ## Extracting columns corresponding to hour and date
    hour = hour
    date = str(date)
    column1 = "Bid curve chart data (Reference time)." + str(hour)

    if hour < 10:
        hour = "0" + str(hour)
    else:
        hour = str(hour)
    column2 = date + " " + hour + ":00:00 +"

    df2 = pd.DataFrame({'name': df[column1],
                        'value' : df[column2]})

    # Extracting net flows
```

```

df2['net_flows'] = df2['value'].where(df2['name']=="Bid curve chart data",
↳(Volume for net flows))
# Forward fill
df2['net_flows'] = df2['net_flows'].ffill(axis = 0)

# Extracting volume for accepted blocks buy
df2['accepted_blocks_buy'] = df2['value'].where(df2['name']=="Bid curve",
↳chart data (Volume for accepted blocks buy))
# Forward fill
df2['accepted_blocks_buy'] = df2['accepted_blocks_buy'].ffill(axis = 0)

# Extracting volume for accepted blocks sell
df2['accepted_blocks_sell'] = df2['value'].where(df2['name']=="Bid curve",
↳chart data (Volume for accepted blocks sell))
# Forward fill
df2['accepted_blocks_sell'] = df2['accepted_blocks_sell'].ffill(axis = 0)

# Identify the rows that have buy or sell "curve"
df2['buy_sell'] = df2['name'].where(df2['name'].str.contains('curve'))
# Forward fill
df2['buy_sell'] = df2['buy_sell'].ffill(axis = 0)

# Extract prices and volumes
df2['price'] = df2['value'][df2['name']=="Price value"]
df2['volume'] = df2['value'][df2['name']=="Volume value"]

# Forward fill prices
df2['price'] = df2['price'].ffill(axis = 0)
# Backward fill volumes
df2['volume'] = df2['volume'].bfill(axis=0)

# Dropping duplicates
df2.drop_duplicates(subset=['price','volume'], keep = 'first', inplace =
↳True)

# Dropping "Buy curve" & "Sell curve" rows
df2 = df2.drop(df2[(df2['name'] == "Buy curve") | (df2['name'] == "Sell",
↳curve)].index)

# Adjusting Buy and Sell volumes
if df2['net_flows'].any() >= 0:
    df2['volume_adjusted'] = np.where(df2['buy_sell']=="Buy",
↳curve",df2['volume']+df2['accepted_blocks_buy'],df2['volume'])
    df2['volume_adjusted'] = np.where(df2['buy_sell']=="Sell",
↳curve",df2['volume']+df2['accepted_blocks_sell']+df2['net_flows'],df2['volume_adjusted'])
if df2['net_flows'].any() < 0:

```

```

df2['volume_adjusted'] = np.where(df2['buy_sell']=="Buy_
→curve",df2['volume']+df2['accepted_blocks_buy']+df2['net_flows'],df2['volume'])
df2['volume_adjusted'] = np.where(df2['buy_sell']=="Sell_
→curve",df2['volume']+df2['accepted_blocks_sell'],df2['volume_adjusted'])

# Keeping relevant data
df2 = df2[['buy_sell','price','volume', 'volume_adjusted',
→'accepted_blocks_buy', 'accepted_blocks_sell', 'net_flows']]
df2 = df2.dropna()

# Basic stats
#print(df2.groupby('buy_sell').describe())

return df2, date, hour

```

```

[4]: file = "/Users/marcosdemetry/Dropbox/IFN/Programming/Python_projects/
→elspot_supply_demand/mcp_data_report_20-08-2020-00_00_00.xls.xlsm"
df = pd.read_excel(file)

```

```

[5]: df2, date, hour = cleaningData("20.08.2020",8)
df2.groupby('buy_sell').price.describe()

```

```

[5]:
count      mean      std      min      25%      50%      75%  \
buy_sell
Buy curve  682.0  127.526222  405.532865 -500.0  0.855265  37.6   87.275
Sell curve  873.0  447.120121  930.264224 -500.0  2.466325  35.5  239.000

max
buy_sell
Buy curve  3000.0
Sell curve  3000.0

```

```

[6]: df2.groupby('buy_sell').volume_adjusted.describe()

```

```

[6]:
count      mean      std      min      25%  \
buy_sell
Buy curve  682.0  34632.76116  1463.236695  32829.700000  33573.868156
Sell curve  873.0  35377.61061  10655.943005  16499.840339  25793.393289

50%      75%      max
buy_sell
Buy curve  34319.122409  35430.024514  39479.325070
Sell curve  40868.556405  43132.378639  47288.049601

```

1.2 Creating figure

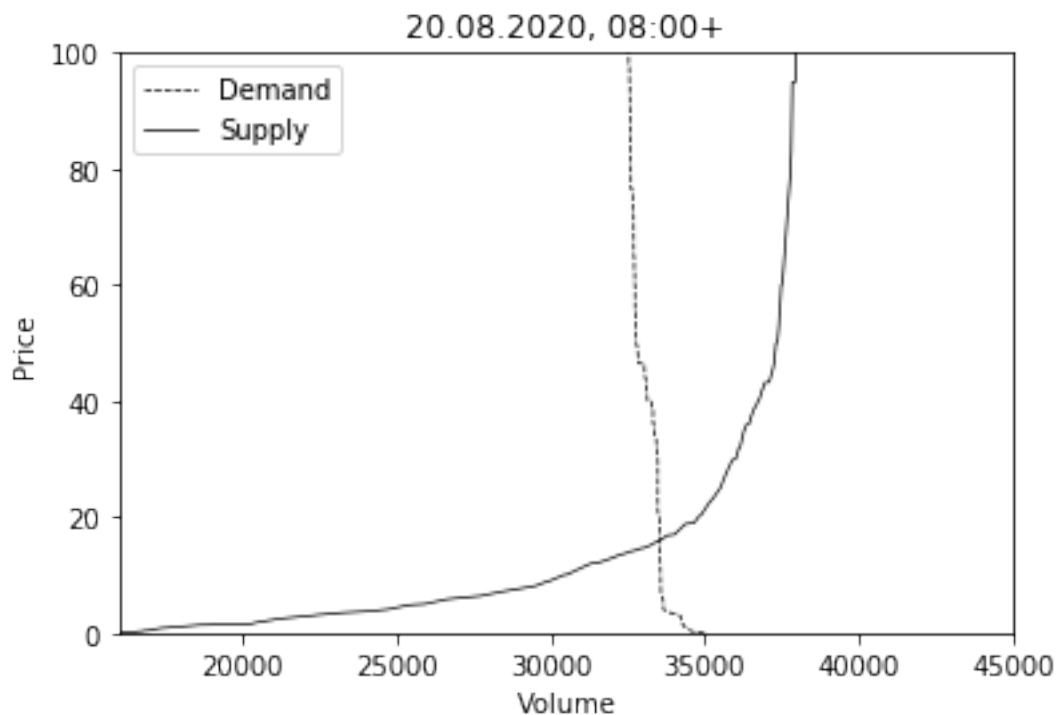
```
[7]: from matplotlib import pyplot as plt
    %matplotlib inline

[8]: y_demand = df2['price'][df2['buy_sell']=="Buy curve"]
    y_supply = df2['price'][df2['buy_sell']=="Sell curve"]

    x_demand = df2['volume'][df2['buy_sell']=="Buy curve"]
    x_supply = df2['volume'][df2['buy_sell']=="Sell curve"]

    demand = plt.plot(x_demand, y_demand, 'k--', label= "Demand", linewidth=0.7)
    supply = plt.plot(x_supply, y_supply, 'k-', label= "Supply", linewidth=0.7)
    plt.ylabel('Price')
    plt.xlabel('Volume')
    plt.ylim((0, 100))
    plt.xlim((16000,45000))
    title = str(date) + ", " + str(hour) + ":00+"
    plt.title(title)
    plt.legend()

    date = date.replace(".", "_")
    outfile = "/Users/marcosdemetry/Dropbox/IFN/Programming/Python_projects/
    ↳elspot_supply_demand/Figure_" + str(date) + "_hr" + str(hour) + ".pdf"
    plt.savefig(outfile, bbox_inches='tight')
```



1.3 Exporting data

```
[9]: outfile = "/Users/marcosdemetry/Dropbox/IFN/Programming/Python_projects/  
      ↳ elspot_supply_demand/Data_" + str(date) + "_hr" + str(hour) + ".xlsx"  
      df2.to_excel(outfile, index=False)
```