Coin Metrics Options Data Aggregation

This notebook demonstrates basic functionality offered by the Coin Metrics Python API Client and Market Data Feed.

Coin Metrics offers a vast assortment of data for hundreds of cryptoassets. The Python API Client allows for easy access to this data using Python without needing to create your own wrappers using `requests` and other such libraries.

Resources

To understand the data that Coin Metrics offers, feel free to peruse the resources below.

- The [Coin Metrics API v4 (https://docs.coinmetrics.io/api/v4)](https://docs.coinmetrics.io/api/v4) website contains the full set of endpoints and data offered by Coin Metrics.
- The [Coin Metrics Knowledge Base (https://docs.coinmetrics.io/info)](https://docs.coinmetrics.io/info) gives detailed, conceptual explanations of the data that Coin Metrics offers.
- The [API Spec (https://coinmetrics.github.io/api-client-python/site/api_client.html)](https://coinmetrics.github.io/api-client-python/site/api_client.html) contains a full list of functions.

Notebook Setup

In [1]:

```
import os
from os import environ
import sys
import pandas as pd
import numpy as np
import seaborn as sns
import logging
from datetime import date, datetime, timedelta
from coinmetrics.api_client import CoinMetricsClient
import json
import logging
from datetime import timezone as timezone_info
import matplotlib.ticker as mticker
from matplotlib.ticker import ScalarFormatter
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
%matplotlib inline
from plotly import graph_objects as go
```

executed in 611ms, finished 09:57:24 2022-11-16

In [2]:

```
sns.set_theme()
sns.set(rc={'figure.figsize':(14,8)})
sns.set_style("whitegrid",{'axes.grid' : True,'grid.linestyle': '--', 'grid.color': 'gray','axes.edgecolor': 'white','font.fa
```

executed in 5ms, finished 09:57:24 2022-11-16

In [3]:

```
logging.basicConfig(
    format='%(asctime)s %(levelname)-8s %(message)s',
    level=logging.INFO,
    datefmt='%Y-%m-%d %H:%M:%S'
)
```

executed in 9ms, finished 09:57:24 2022-11-16

In [4]:

```
# We recommend privately storing your API key in your local environment.
try:
    api_key = environ["CM_API_KEY"]
    logging.info("Using API key found in environment")
except KeyError:
    api_key = ""
    logging.info("API key not found. Using community client")

client = CoinMetricsClient(api_key)
```

executed in 8ms, finished 09:57:24 2022-11-16

```
2022-11-16 14:57:24 INFO     Using API key found in environment
```

Set start and end time

In [5]:

```
end = datetime.utcnow()
start = '2022-08-15'
end = '2022-11-16'
```

executed in 8ms, finished 09:57:24 2022-11-16

OPTIONS CATALOG

To simplify some of the complexity around options contract size, we will investigate only options markets on Deribit, the largest options exchange by volume.

In [6]:
```python
market_catalog = client.catalog_markets(
    market_type='option',
    exchange='deribit'
)
```
executed in 1.27s, finished 09:57:25 2022-11-16

In [7]:
```python
print('Total number of supported options markets: ' + str(len(market_catalog)))
```
executed in 8ms, finished 09:57:25 2022-11-16

```
Total number of supported options markets: 55188
```

In [8]:
```python
market_catalog[1]
```
executed in 11ms, finished 09:57:25 2022-11-16

Out[8]:

```
{'market': 'deribit-BTC-10APR21-52000-P-option',
 'min_time': '2021-04-08T16:11:48.085000000Z',
 'max_time': '2021-04-08T22:10:47.299000000Z',
 'exchange': 'deribit',
 'type': 'option',
 'trades': {'min_time': '2021-04-08T16:11:48.085000000Z',
  'max_time': '2021-04-08T22:10:47.299000000Z'},
 'base': 'btc',
 'quote': 'usd',
 'symbol': 'BTC-10APR21-52000-P',
 'size_asset': 'btc',
 'strike': '52000',
 'option_contract_type': 'put',
 'is_european': True,
 'contract_size': '1',
 'listing': '2021-04-08T08:00:06.000000000Z',
 'expiration': '2021-04-10T08:00:00.000000000Z',
 'settlement_price': '60483.68'}
```


Filter for target asset

In [9]:
```python
asset = 'eth'
```
executed in 7ms, finished 09:57:25 2022-11-16

In [10]:
```python
asset_options = []
try:
    for item in market_catalog:
        if (item['base']==asset):
            asset_options.append(item)
except KeyError:
        pass
```
executed in 38ms, finished 09:57:25 2022-11-16

In [11]:
```python
print('Total number of supported ' + str(asset).upper() + ' options markets: ' + str(len(asset_options)))
```
executed in 3ms, finished 09:57:25 2022-11-16

```
Total number of supported ETH options markets: 26178
```

Filter out expired/inactive contracts

In [12]:

```
asset_options = pd.DataFrame(asset_options)
asset_options['max_time'] = pd.to_datetime(asset_options['max_time'])
asset_options = asset_options.loc[(asset_options.max_time > start)]
asset_options
```

executed in 213ms, finished 09:57:26 2022-11-16

Out[12]:

	market	min_time	max_time	exchange	type	trades	base	quote	symbol	size_asset	strike	option_contract_type	i
631	deribit-ETH-10NOV22-1000-C-option	2022-11-08T08:01:00.000000000Z	2022-11-10 07:59:00+00:00	deribit	option	NaN	eth	eth	ETH-10NOV22-1000-C	eth	1000	call	
632	deribit-ETH-10NOV22-1000-P-option	2022-11-08T08:01:00.000000000Z	2022-11-10 07:59:00+00:00	deribit	option	{'min_time': '2022-11-08T15:40:21.913000000Z',...	eth	eth	ETH-10NOV22-1000-P	eth	1000	put	
633	deribit-ETH-10NOV22-1050-C-option	2022-11-09T13:19:00.000000000Z	2022-11-10 07:59:00+00:00	deribit	option	{'min_time': '2022-11-09T21:54:12.078000000Z',...	eth	eth	ETH-10NOV22-1050-C	eth	1050	call	
634	deribit-ETH-10NOV22-1050-P-option	2022-11-09T13:19:00.000000000Z	2022-11-10 07:59:00+00:00	deribit	option	{'min_time': '2022-11-09T13:52:15.543000000Z',...	eth	eth	ETH-10NOV22-1050-P	eth	1050	put	
635	deribit-ETH-10NOV22-1100-C-option	2022-11-08T08:01:00.000000000Z	2022-11-10 07:59:00+00:00	deribit	option	{'min_time': '2022-11-09T08:45:14.513000000Z',...	eth	eth	ETH-10NOV22-1100-C	eth	1100	call	
...	...	...	...	...	...	...	...	...	...	...	...	...	
26173	deribit-ETH-9SEP22-3200-P-option	2022-08-18T08:01:00.000000000Z	2022-09-09 07:59:00+00:00	deribit	option	NaN	eth	eth	ETH-9SEP22-3200-P	eth	3200	put	
26174	deribit-ETH-9SEP22-3400-C-option	2022-08-18T08:01:00.000000000Z	2022-09-09 07:59:00+00:00	deribit	option	{'min_time': '2022-08-20T17:28:27.531000000Z',...	eth	eth	ETH-9SEP22-3400-C	eth	3400	call	
26175	deribit-ETH-9SEP22-3400-P-option	2022-08-18T08:01:00.000000000Z	2022-09-09 07:59:00+00:00	deribit	option	NaN	eth	eth	ETH-9SEP22-3400-P	eth	3400	put	
26176	deribit-ETH-9SEP22-800-C-option	2022-08-18T08:01:00.000000000Z	2022-09-09 07:59:00+00:00	deribit	option	NaN	eth	eth	ETH-9SEP22-800-C	eth	800	call	
26177	deribit-ETH-9SEP22-800-P-option	2022-08-18T08:01:00.000000000Z	2022-09-09 07:59:00+00:00	deribit	option	{'min_time': '2022-08-18T12:45:54.184000000Z',...	eth	eth	ETH-9SEP22-800-P	eth	800	put	

4210 rows × 18 columns

TRADE VOLUME

On Deribit, each option contract provides notional exposure to 1 unit of the underlying asset (i.e. 1 BTC or 1 ETH). To calculate options volume, we must multiply the 'amount' field (representing the number of contracts traded) by the USD price of the underlying asset.

In [13]:

```
# Drop markets without trades data
trades_cat = asset_options.dropna(subset=['trades'])
trades_cat = trades_cat['market'].tolist()
len(trades_cat)
```

executed in 14ms, finished 09:57:26 2022-11-16

Out[13]: 2746

In [14]:

```
full_vol = pd.DataFrame()
batch_size = 420

for i in range(0, len(trades_cat),batch_size):
    mkt_batch = trades_cat[i:i+batch_size]
    print(str('Retrieving batch of trades for markets ' + str(i) + ' - ' + str(i+batch_size) + '...'))

    vol_batch = client.get_market_trades(
        markets=mkt_batch,
        start_time = start,
        end_time= end
    ).to_dataframe()
    print('Retrieved batch of ' + str(len(vol_batch)) + ' trades')

    full_vol = pd.concat((full_vol, vol_batch), axis = 0, ignore_index=False)
    print('Total of ' + str(len(full_vol)) + ' trades\n')

full_vol = full_vol.reset_index(drop=True)
```

executed in 43m 22s, finished 10:40:48 2022-11-16

```
Retrieving batch of trades for markets 0 - 420...
Retrieved batch of 151752 trades
Total of 151752 trades

Retrieving batch of trades for markets 420 - 840...
Retrieved batch of 129210 trades
Total of 280962 trades

Retrieving batch of trades for markets 840 - 1260...
Retrieved batch of 237874 trades
Total of 518836 trades

Retrieving batch of trades for markets 1260 - 1680...
Retrieved batch of 219154 trades
Total of 737990 trades

Retrieving batch of trades for markets 1680 - 2100...
Retrieved batch of 265882 trades
Total of 1003872 trades

Retrieving batch of trades for markets 2100 - 2520...
Retrieved batch of 116484 trades
Total of 1120356 trades

Retrieving batch of trades for markets 2520 - 2940...
Retrieved batch of 92721 trades
Total of 1213077 trades
```

In [15]:

```
full_vol = full_vol[['market','time','amount']]
full_vol = full_vol[~(full_vol['time'] > end)]
full_vol.sort_values('time')
```

executed in 226ms, finished 10:40:48 2022-11-16

Out[15]:

	market	time	amount
268555	deribit-ETH-19AUG22-1900-P-option	2022-08-15 00:00:59.498000+00:00	1
270571	deribit-ETH-19AUG22-2200-C-option	2022-08-15 00:02:55.944000+00:00	14
721934	deribit-ETH-28OCT22-4000-C-option	2022-08-15 00:03:01.071000+00:00	1
721935	deribit-ETH-28OCT22-4000-C-option	2022-08-15 00:03:01.502000+00:00	2
268556	deribit-ETH-19AUG22-1900-P-option	2022-08-15 00:05:21.805000+00:00	10
...	...	...	...
210022	deribit-ETH-17NOV22-1325-C-option	2022-11-15 23:58:11.542000+00:00	1
444260	deribit-ETH-25NOV22-1350-P-option	2022-11-15 23:59:07.589000+00:00	1
444261	deribit-ETH-25NOV22-1350-P-option	2022-11-15 23:59:07.628000+00:00	2
164922	deribit-ETH-16NOV22-1250-P-option	2022-11-15 23:59:08.106000+00:00	2
165899	deribit-ETH-16NOV22-1300-C-option	2022-11-15 23:59:50.731000+00:00	1

1203624 rows × 3 columns

Retrieve 1-minute Reference Rate

In [16]:

```
df_refrate = client.get_asset_metrics(
    assets=asset,
    frequency='1m',
    metrics='ReferenceRateUSD',
    start_time=start,
    end_time=end
).to_dataframe()
```

executed in 4m 11s, finished 10:44:59 2022-11-16

In [17]:
```
df_refrate = df_refrate.set_index('time').sort_index().drop(columns=['asset'])
df_refrate
```
executed in 18ms, finished 10:44:59 2022-11-16

Out[17]:

	ReferenceRateUSD
time	
2022-08-15 00:00:00+00:00	1935.85
2022-08-15 00:01:00+00:00	1935.3
2022-08-15 00:02:00+00:00	1935.61
2022-08-15 00:03:00+00:00	1933.43
2022-08-15 00:04:00+00:00	1934.18
...	...
2022-11-16 15:40:00+00:00	1207.33
2022-11-16 15:41:00+00:00	1206.59
2022-11-16 15:42:00+00:00	1205.08
2022-11-16 15:43:00+00:00	1204.94
2022-11-16 15:44:00+00:00	1203.9

134865 rows × 1 columns

In [18]:
```
full_vol['time'] = full_vol['time'].round('T')

# Merge trades data with 1s reference rate
full_vol = full_vol.merge(df_refrate.reset_index(),how='left', on='time')
```
executed in 257ms, finished 10:44:59 2022-11-16

Calculate USD value of trades

In [19]:
```
full_vol['USD Volume'] = full_vol['amount'] * full_vol['ReferenceRateUSD']
```
executed in 7ms, finished 10:44:59 2022-11-16

In [20]:
```
full_vol
```
executed in 16ms, finished 10:44:59 2022-11-16

Out[20]:

	market	time	amount	ReferenceRateUSD	USD Volume
0	deribit-ETH-10NOV22-1000-P-option	2022-11-08 15:40:00+00:00	5	1439.47	7197.35
1	deribit-ETH-10NOV22-1000-P-option	2022-11-08 18:25:00+00:00	4	1381.73	5526.92
2	deribit-ETH-10NOV22-1000-P-option	2022-11-08 18:25:00+00:00	34	1381.73	46978.82
3	deribit-ETH-10NOV22-1000-P-option	2022-11-08 18:25:00+00:00	1	1381.73	1381.73
4	deribit-ETH-10NOV22-1000-P-option	2022-11-08 18:25:00+00:00	1	1381.73	1381.73
...	...	...	...	...	...
1203619	deribit-ETH-9SEP22-800-P-option	2022-08-31 07:04:00+00:00	5	1585.04	7925.2
1203620	deribit-ETH-9SEP22-800-P-option	2022-09-01 08:02:00+00:00	1	1542.06	1542.06
1203621	deribit-ETH-9SEP22-800-P-option	2022-09-01 15:02:00+00:00	1	1515.53	1515.53
1203622	deribit-ETH-9SEP22-800-P-option	2022-09-03 13:54:00+00:00	10	1563.22	15632.2
1203623	deribit-ETH-9SEP22-800-P-option	2022-09-03 13:54:00+00:00	4	1563.22	6252.88

1203624 rows × 5 columns

In [39]:
```
vol_sum = full_vol.groupby(pd.Grouper(key='time', axis=0, freq='1D', sort=True)).sum()
```
executed in 339ms, finished 11:08:18 2022-11-16

In [40]:
```
vol_sum.to_csv('./deribit_' + str(asset).upper() + '_options_vol_' + str(start) + '_to_' + str(end) + '.csv')
```
executed in 5ms, finished 11:08:18 2022-11-16

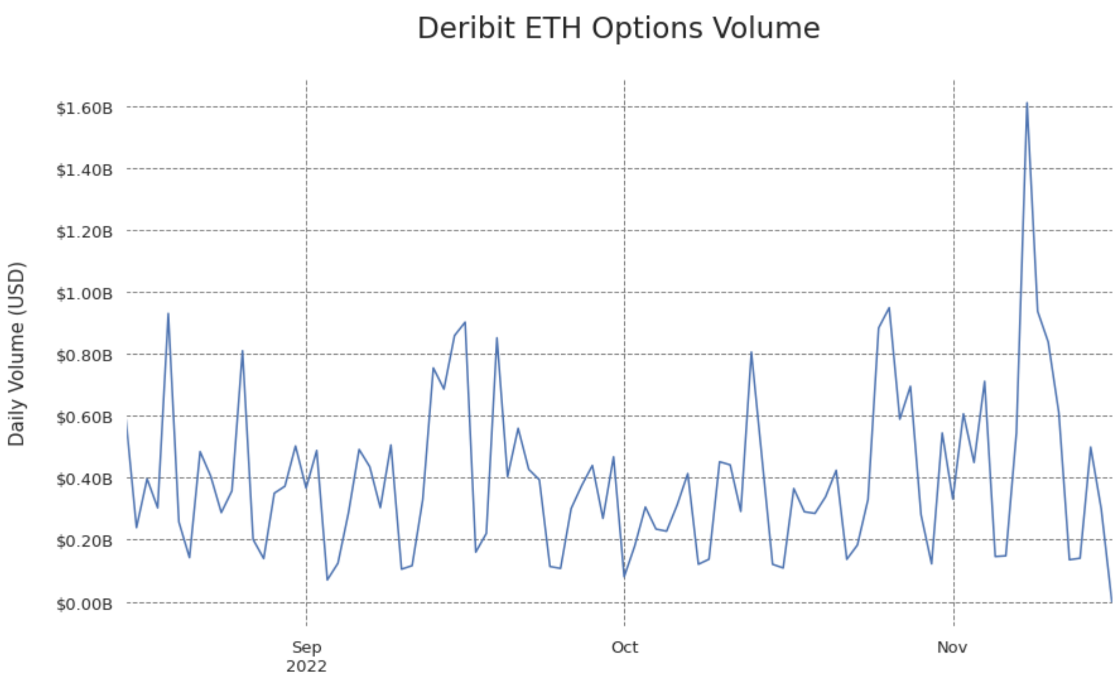
In [41]:
```
vol_sum[['USD Volume']]
```
executed in 10ms, finished 11:08:18 2022-11-16

Out[41]:

	USD Volume
time	
2022-08-15 00:00:00+00:00	589538668.911003
2022-08-16 00:00:00+00:00	238814703.724999
2022-08-17 00:00:00+00:00	396383713.160999
2022-08-18 00:00:00+00:00	302514426.043001
2022-08-19 00:00:00+00:00	929990961.332017
...	...
2022-11-12 00:00:00+00:00	135147139.310001
2022-11-13 00:00:00+00:00	140129443.769999
2022-11-14 00:00:00+00:00	498790211.079997
2022-11-15 00:00:00+00:00	299006875.790001
2022-11-16 00:00:00+00:00	1251.74

94 rows × 1 columns

In [42]:
```
v = vol_sum.plot.line(y='USD Volume')
v.set_xlabel("Date", fontsize = 15)
plt.setp(v.get_yticklabels(), fontsize=13)
plt.setp(v.get_xticklabels(), fontsize=13)
v.set_xlabel("")
plt.xlim([vol_sum.index[0], vol_sum.index[-1]])
v.set_ylabel("Daily Volume (USD)\n", fontsize = 15)
v.set_title(('\nDeribit ' + str(asset).upper() + ' Options Volume\n'),fontsize=23)
v.get_legend().remove()
v.yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, pos: '${:,.2f}'.format(x/1000000000) + 'B'))
```
executed in 348ms, finished 11:08:19 2022-11-16



OPEN INTEREST

Open interest represents the number of contracts that are currently outstanding and not settled for a specific derivatives market.

In [25]:
```python
# Drop markets without open interest data
oi_cat = asset_options.dropna(subset=['openinterest'])
oi_cat['listing'] = pd.to_datetime(asset_options['listing'])
oi_cat = oi_cat.loc[(oi_cat.listing < end)]
oi_cat = oi_cat['market'].tolist()
len(oi_cat)
```
executed in 19ms, finished 10:45:00 2022-11-16

Out[25]: 4206

In [26]:
```python
full_oi = pd.DataFrame()
batch_size = 420

for i in range(0, len(oi_cat),batch_size):
    mkt_batch = oi_cat[i:i+batch_size]
    print(str('\nRetrieving EOD open interest for markets ' + str(i) + ' through ' + str(i+batch_size) + '...'))

    start_date = pd.to_datetime(start)
    end_date = pd.to_datetime(end)
    delta = timedelta(days=1)
    print(str(start_date) + '\n...')

    while start_date <= end_date:

        oi_batch = client.get_market_open_interest(
            markets=mkt_batch,
            start_time = start_date.strftime("%Y-%m-%d"),
            limit_per_market = 1,
            paging_from='start',
            timezone='America/New_York'
        ).to_dataframe()
        #print(oi_batch)
        full_oi = pd.concat((full_oi, oi_batch), axis = 0, ignore_index=False)
        start_date += delta

    print(start_date)
```
executed in 19m 22s, finished 11:04:22 2022-11-16

```
Retrieving EOD open interest for markets 0 through 420...
2022-08-15 00:00:00
...

2022-11-16 15:47:09 INFO     no data to export
2022-11-16 15:47:09 WARNING  Response is empty.
2022-11-16 15:47:10 INFO     no data to export
2022-11-16 15:47:10 WARNING  Response is empty.
2022-11-16 15:47:10 INFO     no data to export
2022-11-16 15:47:10 WARNING  Response is empty.

2022-11-17 00:00:00

Retrieving EOD open interest for markets 420 through 840...
2022-08-15 00:00:00
...
2022-11-17 00:00:00

Retrieving EOD open interest for markets 840 through 1260...
```

In [27]:
```python
full_oi
```
executed in 17ms, finished 11:04:22 2022-11-16

Out[27]:

	market	time	contract_count	value_usd	database_time	exchange_time
0	deribit-ETH-10NOV22-1000-C-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:08.918440+00:00	2022-11-08 08:01:00+00:00
1	deribit-ETH-10NOV22-1000-P-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:08.918440+00:00	2022-11-08 08:01:00+00:00
2	deribit-ETH-10NOV22-1050-C-option	2022-11-09 13:19:00+00:00	0	0.0	2022-11-09 13:19:07.139255+00:00	2022-11-09 13:19:00+00:00
3	deribit-ETH-10NOV22-1050-P-option	2022-11-09 13:19:00+00:00	0	0.0	2022-11-09 13:19:07.139255+00:00	2022-11-09 13:19:00+00:00
4	deribit-ETH-10NOV22-1100-C-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:09.917989+00:00	2022-11-08 08:01:00+00:00
...	...	...	...	...	...	...
1	deribit-ETH-9SEP22-3200-P-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:25.451175+00:00	2022-09-09 04:00:00+00:00
2	deribit-ETH-9SEP22-3400-C-option	2022-09-09 04:00:00+00:00	10	16710.5	2022-09-09 04:00:30.451434+00:00	2022-09-09 04:00:00+00:00
3	deribit-ETH-9SEP22-3400-P-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:17.449583+00:00	2022-09-09 04:00:00+00:00
4	deribit-ETH-9SEP22-800-C-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:26.451431+00:00	2022-09-09 04:00:00+00:00
5	deribit-ETH-9SEP22-800-P-option	2022-09-09 04:00:00+00:00	1198	2003595.1	2022-09-09 04:00:20.450512+00:00	2022-09-09 04:00:00+00:00

229080 rows × 6 columns

In [28]:
```python
oi_sum = full_oi.groupby(pd.Grouper(key='time', axis=0, freq='1D', sort=True)).sum()
```
executed in 73ms, finished 11:04:22 2022-11-16

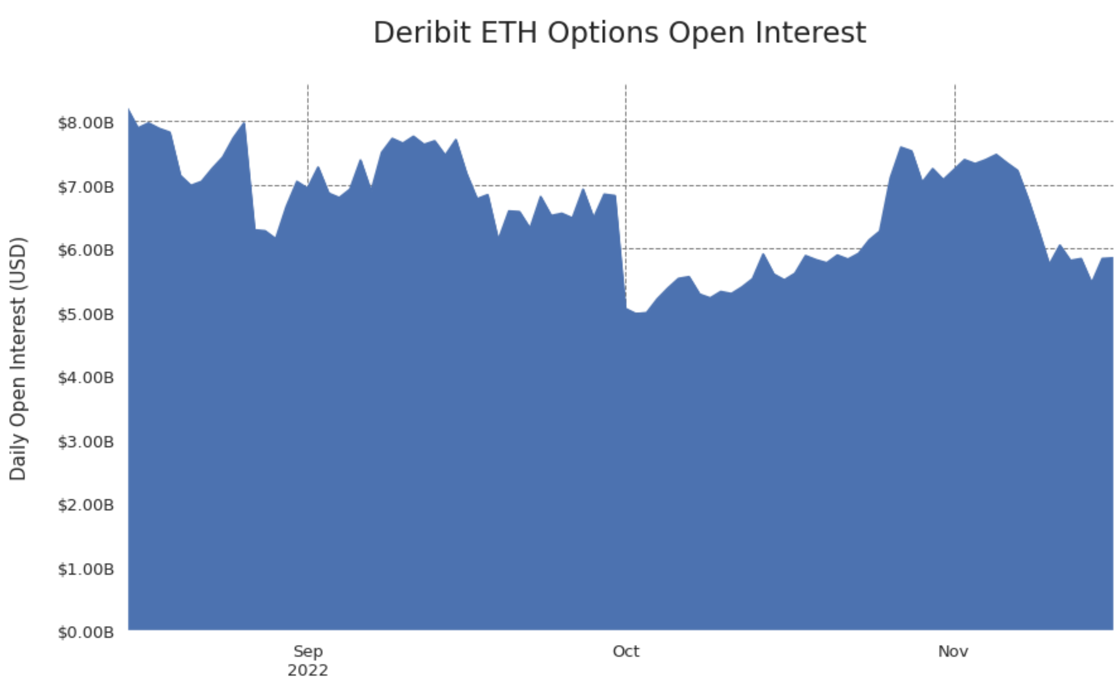
In [29]:
```
oi_sum.to_csv('./deribit_' + str(asset).upper() + '_options_oi_' + str(start) + '_to_' + str(end) + '.csv')
oi_sum
```
executed in 14ms, finished 11:04:22 2022-11-16

Out[29]:

	contract_count	value_usd
time		
2022-08-15 00:00:00+00:00	4123317	8196194876.509999
2022-08-16 00:00:00+00:00	4177567	7887245633.829995
2022-08-17 00:00:00+00:00	4200431	7966457887.849999
2022-08-18 00:00:00+00:00	4261966	7880240544.19
2022-08-19 00:00:00+00:00	4302673	7821041289.620001
...	...	...
2022-11-12 00:00:00+00:00	4632923	5808514314.160005
2022-11-13 00:00:00+00:00	4617698	5837944795.789996
2022-11-14 00:00:00+00:00	4612170	5454715321.299999
2022-11-15 00:00:00+00:00	4646565	5838600878.9
2022-11-16 00:00:00+00:00	4654832	5849344435.150004

94 rows × 2 columns

In [30]:
```
#p = sns.lineplot(data=vol_sum, x="time", y="USD Volume")
oi = oi_sum.plot.area(y='value_usd')
oi.set_xlabel("Date", fontsize = 15)
plt.setp(oi.get_yticklabels(), fontsize=13)
plt.setp(oi.get_xticklabels(), fontsize=13)
oi.set_xlabel("")
plt.xlim([oi_sum.index[0], oi_sum.index[-1]])
oi.set_ylabel("Daily Open Interest (USD)\n", fontsize = 15)
oi.set_title(('\nDeribit ' + str(asset).upper() + ' Options Open Interest\n'),fontsize=23)
oi.get_legend().remove()
oi.yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, pos: '${:,.2f}'.format(x/1000000000) + 'B'))
```
executed in 355ms, finished 11:04:23 2022-11-16



Split into Calls/Puts

In [31]:
```
full_oi_split = full_oi.merge(asset_options[['market','option_contract_type']],how='left', on='market')
```
executed in 155ms, finished 11:04:23 2022-11-16

In [32]:
```
full_oi_split
```
executed in 19ms, finished 11:04:23 2022-11-16

Out[32]:

	market	time	contract_count	value_usd	database_time	exchange_time	option_contract_type
0	deribit-ETH-10NOV22-1000-C-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:08.918440+00:00	2022-11-08 08:01:00+00:00	call
1	deribit-ETH-10NOV22-1000-P-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:08.918440+00:00	2022-11-08 08:01:00+00:00	put
2	deribit-ETH-10NOV22-1050-C-option	2022-11-09 13:19:00+00:00	0	0.0	2022-11-09 13:19:07.139255+00:00	2022-11-09 13:19:00+00:00	call
3	deribit-ETH-10NOV22-1050-P-option	2022-11-09 13:19:00+00:00	0	0.0	2022-11-09 13:19:07.139255+00:00	2022-11-09 13:19:00+00:00	put
4	deribit-ETH-10NOV22-1100-C-option	2022-11-08 08:01:00+00:00	0	0.0	2022-11-08 08:01:09.917989+00:00	2022-11-08 08:01:00+00:00	call
...	...	...	...	...	...	...	...
229075	deribit-ETH-9SEP22-3200-P-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:25.451175+00:00	2022-09-09 04:00:00+00:00	put
229076	deribit-ETH-9SEP22-3400-C-option	2022-09-09 04:00:00+00:00	10	16710.5	2022-09-09 04:00:30.451434+00:00	2022-09-09 04:00:00+00:00	call
229077	deribit-ETH-9SEP22-3400-P-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:17.449583+00:00	2022-09-09 04:00:00+00:00	put
229078	deribit-ETH-9SEP22-800-C-option	2022-09-09 04:00:00+00:00	0	0.0	2022-09-09 04:00:26.451431+00:00	2022-09-09 04:00:00+00:00	call
229079	deribit-ETH-9SEP22-800-P-option	2022-09-09 04:00:00+00:00	1198	2003595.1	2022-09-09 04:00:20.450512+00:00	2022-09-09 04:00:00+00:00	put

229080 rows × 7 columns

In [33]:
```
calls = full_oi_split.loc[(full_oi_split.option_contract_type == 'call')]
call_sum_split = calls.groupby(pd.Grouper(key='time', axis=0, freq='1D', sort=True)).sum()
puts = full_oi_split.loc[(full_oi_split.option_contract_type == 'put')]
put_sum_split = puts.groupby(pd.Grouper(key='time', axis=0, freq='1D', sort=True)).sum()
```
executed in 148ms, finished 11:04:23 2022-11-16

In [34]:
```
put_sum_split = put_sum_split.rename(columns={"value_usd": "Puts - USD Value", "contract_count": "Puts - Contract Count" })
call_sum_split = call_sum_split.rename(columns={"value_usd": "Calls - USD Value", "contract_count": "Calls - Contract Count"
```
executed in 3ms, finished 11:04:23 2022-11-16

In [35]:
```
full_oi_split = call_sum_split.merge(put_sum_split,how='left', on='time')
full_oi_split
```
executed in 23ms, finished 11:04:23 2022-11-16

Out[35]:

time	Calls - Contract Count	Calls - USD Value	Puts - Contract Count	Puts - USD Value
2022-08-15 00:00:00+00:00	3293353	6546440376.879999	829964	1649754499.63
2022-08-16 00:00:00+00:00	3325066	6277724583.430001	852501	1609521050.4
2022-08-17 00:00:00+00:00	3335227	6325524552.950001	865204	1640933334.9
2022-08-18 00:00:00+00:00	3364024	6219984505.630005	897942	1660256038.560001
2022-08-19 00:00:00+00:00	3372699	6130582995.060002	929974	1690458294.56
...	...	...	...	...
2022-11-12 00:00:00+00:00	3520320	4413586155.219999	1112603	1394928158.940001
2022-11-13 00:00:00+00:00	3519091	4449018551.230001	1098607	1388926244.559999
2022-11-14 00:00:00+00:00	3526077	4170219796.46	1086093	1284495524.839999
2022-11-15 00:00:00+00:00	3542648	4451493658.979997	1103917	1387107219.919999
2022-11-16 00:00:00+00:00	3559840	4473366890.560002	1094992	1375977544.59

94 rows × 4 columns

In [36]:

```
full_oi_split['Put/Call Ratio'] = full_oi_split['Puts - USD Value'] / full_oi_split['Calls - USD Value']
full_oi_split[['Put/Call Ratio']]
```

executed in 12ms, finished 11:04:23 2022-11-16

Out[36]:

	Put/Call Ratio
time	
2022-08-15 00:00:00+00:00	0.252008
2022-08-16 00:00:00+00:00	0.256386
2022-08-17 00:00:00+00:00	0.259415
2022-08-18 00:00:00+00:00	0.266923
2022-08-19 00:00:00+00:00	0.275742
...	...
2022-11-12 00:00:00+00:00	0.316053
2022-11-13 00:00:00+00:00	0.312187
2022-11-14 00:00:00+00:00	0.308016
2022-11-15 00:00:00+00:00	0.311605
2022-11-16 00:00:00+00:00	0.307593

94 rows × 1 columns

In [37]:

```
full_oi_split.to_csv('./deribit_' + str(asset).upper() + '_options_oi_calls_puts_' + str(start) + '_to_' + str(end) + '.csv')
```

executed in 10ms, finished 11:04:23 2022-11-16

In [38]:

```
pc = sns.lineplot(data=full_oi_split[['Put/Call Ratio']], x="time", y="Put/Call Ratio")
pc.set_xlabel("Date", fontsize = 15)
plt.setp(pc.get_yticklabels(), fontsize=13)
plt.setp(pc.get_xticklabels(), fontsize=13)
pc.set_xlabel("")
plt.xlim([full_oi_split.index[0], full_oi_split.index[-1]])
pc.set_ylabel("Put/Call Ratio\n", fontsize = 15)
pc.set_title(('\nDeribit ' + str(asset).upper() + ' Options\nPut/Call Ratio\n'),fontsize=23)
pc.yaxis.set_major_formatter(mticker.FuncFormatter(lambda x, pos: '{:,.2f}'.format(x)))
```

executed in 423ms, finished 11:04:23 2022-11-16