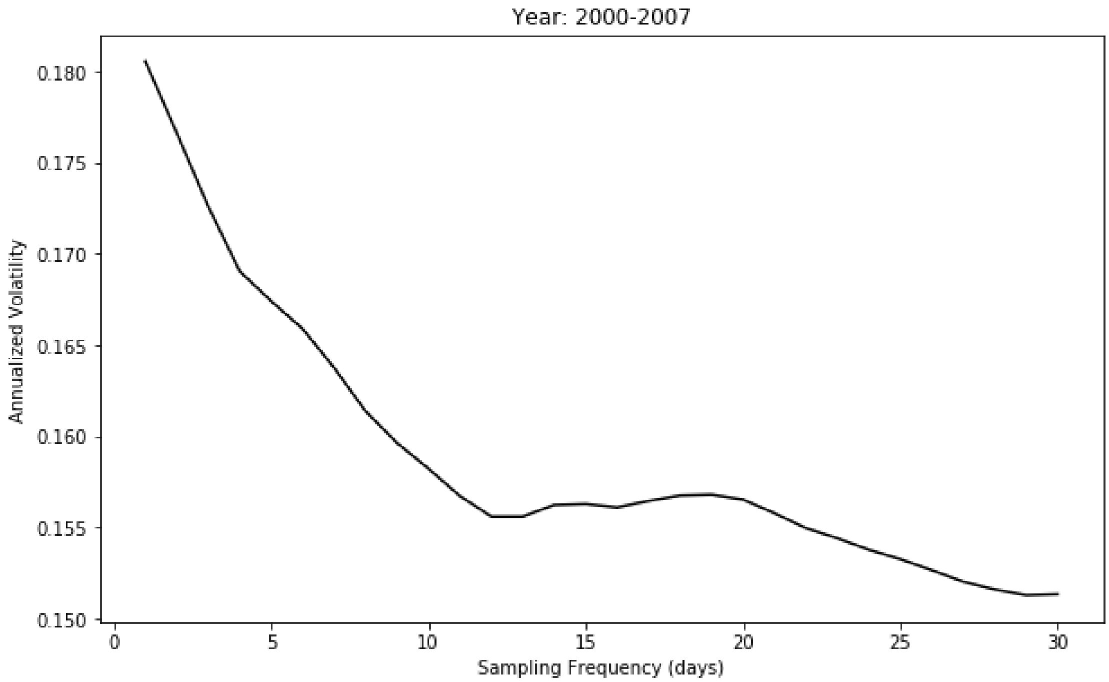
Assignment 1

September 25, 2019

0.1 Annualized Volatility with Different Sampling Frequency (Daily Data)

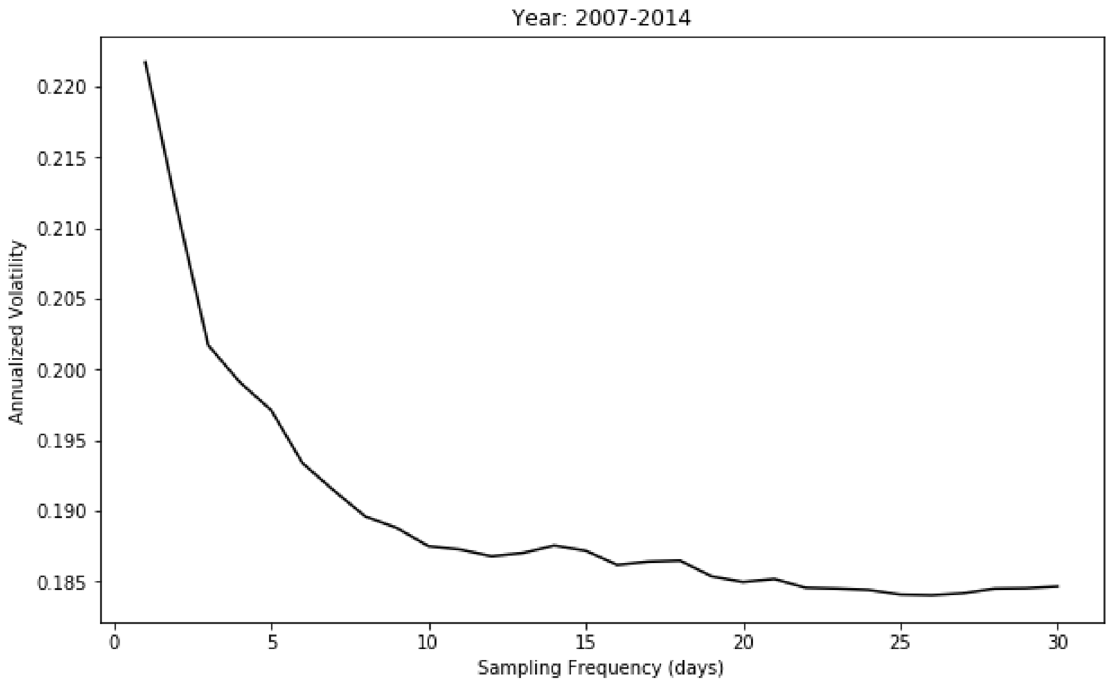
```
In [22]: import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import yfinance as yf

In [23]: def Calculate_Annualized_Volatility(data, shift_period):
             df = data.copy(deep=True)
             log_ret = np.log(df['Close']) - np.log(df['Close'].shift(shift_period))
             log_ret.dropna(inplace=True)

             annualized_vol = np.sqrt(np.var(log_ret)) * np.sqrt(252 / shift_period)
             return annualized_vol

In [24]: def Calculate_Stats(data, shift_period):
             df = data.copy(deep=True)
             log_ret = np.log(df['Close']) - np.log(df['Close'].shift(shift_period))
             log_ret.dropna(inplace=True)

             mean = log_ret.mean() * (252/shift_period)
             median = log_ret.median() * (252/shift_period)
             quantile_25 = np.quantile(log_ret, 0.25) * (252/shift_period)
             quantile_75 = np.quantile(log_ret, 0.75) * (252/shift_period)

             return mean, median, quantile_25, quantile_75

In [27]: spy = yf.Ticker('SPY')
         df = spy.history(start='2000-01-01', end='2007-12-31', interval='1d')

         AnnualizedVol = []
         shift_period = list(range(1,31))
         for i in shift_period:
             vol = Calculate_Annualized_Volatility(df, i)
             AnnualizedVol.append(vol)

In [28]: plt.figure(figsize=(10, 6))
         _ = plt.plot(shift_period, AnnualizedVol, 'k')
```

```
_ = plt.xlabel('Sampling Frequency (days)')
_ = plt.ylabel('Annualized Volatility')
_ = plt.title('Year: 2000-2007')
```




```
In [30]: df = spy.history(start='2007-01-01', end='2014-12-31', interval='1d')

         AnnualizedVol = []
         shift_period = list(range(1,31))
         for i in shift_period:
             vol = Calculate_Annualized_Volatility(df, i)
             AnnualizedVol.append(vol)
```

```
In [31]: plt.figure(figsize=(10, 6))
         _ = plt.plot(shift_period, AnnualizedVol, 'k')
         _ = plt.xlabel('Sampling Frequency (days)')
         _ = plt.ylabel('Annualized Volatility')
         _ = plt.title('Year: 2007-2014')
```

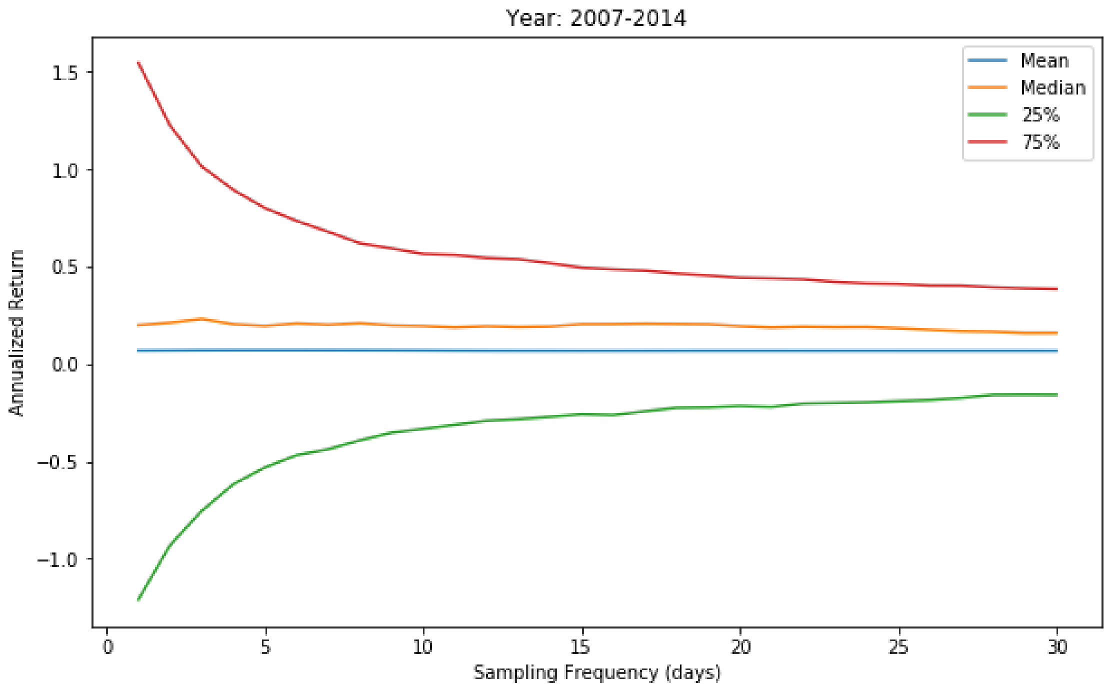


0.2 compute the mean, the median, the 25% and 75% quantile (Daily Data)

```
In [32]: df = spy.history(start='2007-01-01', end='2014-12-31', interval='1d')
```

```
In [33]: means, medians, quantile_25s, quantile_75s = [], [], [], []
         for i in shift_period:
             mean, median, quantile_25, quantile_75 = Calculate_Stats(df, i)
             means.append(mean)
             medians.append(median)
             quantile_25s.append(quantile_25)
             quantile_75s.append(quantile_75)
```

```
In [34]: plt.figure(figsize=(10, 6))
         _ = plt.plot(shift_period, means, label='Mean')
         _ = plt.plot(shift_period, medians, label='Median')
         _ = plt.plot(shift_period, quantile_25s, label='25%')
         _ = plt.plot(shift_period, quantile_75s, label='75%')
         _ = plt.xlabel('Sampling Frequency (days)')
         _ = plt.ylabel('Annualized Return')
         _ = plt.title('Year: 2007-2014')
         _ = plt.legend()
```



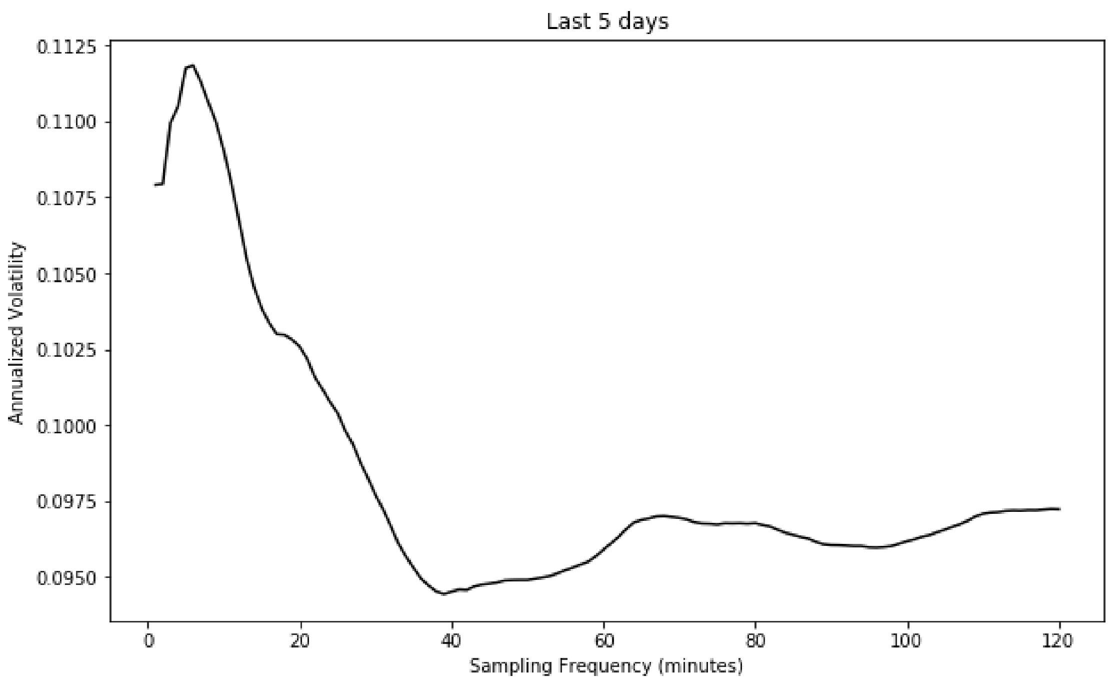

```
In [35]: def Calculate_Annualized_Volatility(data, shift_period):
             df = data.copy(deep=True)
             log_ret = np.log(df['Close']) - np.log(df['Close'].shift(shift_period))
             log_ret.dropna(inplace=True)

             annualized_vol = np.sqrt(np.var(log_ret)) * np.sqrt(252 / shift_period * 6.5*60)
             return annualized_vol

In [36]: def Calculate_Stats(data, shift_period):
             df = data.copy(deep=True)
             log_ret = np.log(df['Close']) - np.log(df['Close'].shift(shift_period))
             log_ret.dropna(inplace=True)

             mean = log_ret.mean() * (252/shift_period) * (6.5*60)
             median = log_ret.median() * (252/shift_period) * (6.5*60)
             quantile_25 = np.quantile(log_ret, 0.25) * (252/shift_period) * (6.5*60)
             quantile_75 = np.quantile(log_ret, 0.75) * (252/shift_period) * (6.5*60)

             return mean, median, quantile_25, quantile_75
```

0.3 Annualized Volatility with Different Sampling Frequency (Intraday Data)

```
In [38]: df = spy.history(period='5d', interval='1m')

         AnnualizedVol = []
```

```
shift_period = list(range(1,121))
for i in shift_period:
    vol = Calculate_Annualized_Volatility(df, i)
    AnnualizedVol.append(vol)
```

```
In [39]: plt.figure(figsize=(10, 6))
         _ = plt.plot(shift_period, AnnualizedVol, 'k')
         _ = plt.xlabel('Sampling Frequency (minutes)')
         _ = plt.ylabel('Annualized Volatility')
         _ = plt.title('Last 5 days')
```

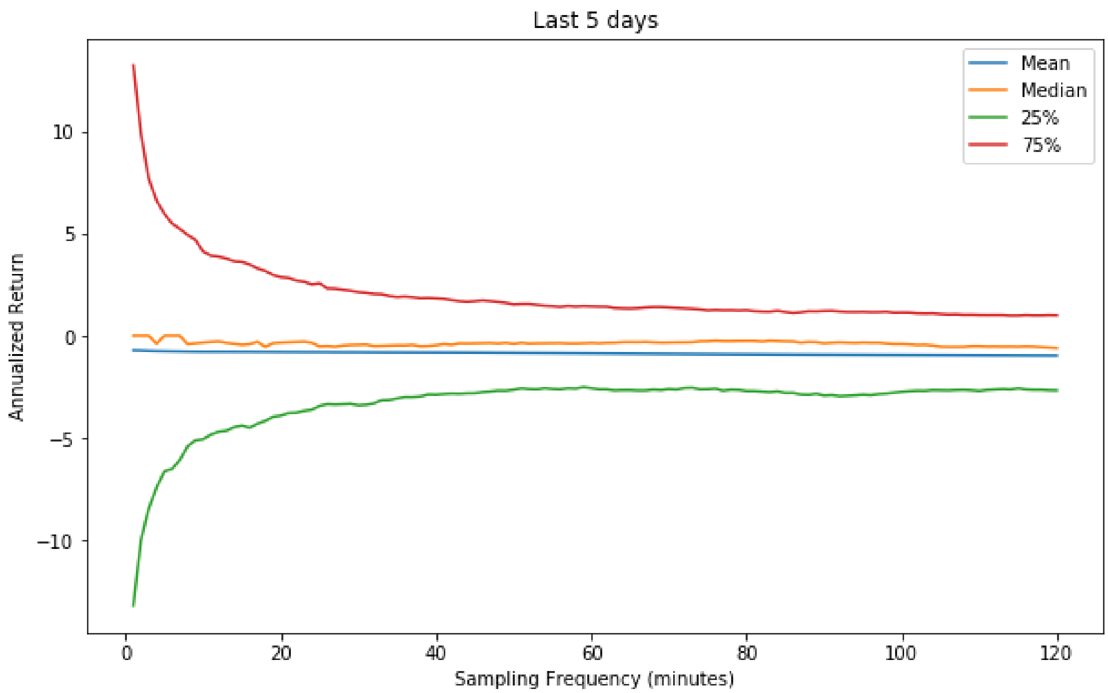


0.4 compute the mean, the median, the 25% and 75% quantile (Intraday Data)

```
In [41]: df = spy.history(period='5d', interval='1m')

         means, medians, quantile_25s, quantile_75s = [], [], [], []
         for i in shift_period:
             mean, median, quantile_25, quantile_75 = Calculate_Stats(df, i)
             means.append(mean)
             medians.append(median)
             quantile_25s.append(quantile_25)
             quantile_75s.append(quantile_75)
```

```
In [42]: plt.figure(figsize=(10, 6))
         _ = plt.plot(shift_period, means, label='Mean')
```

```
_ = plt.plot(shift_period, medians, label='Median')
_ = plt.plot(shift_period, quantile_25s, label='25%')
_ = plt.plot(shift_period, quantile_75s, label='75%')
_ = plt.xlabel('Sampling Frequency (minutes)')
_ = plt.ylabel('Annualized Return')
_ = plt.title('Last 5 days')
_ = plt.legend()
```



0.5 Problem 2 Proof:

Suppose we invested \$a in SPY at the beginning of the first week.

The price of SPY at the begining is $S_{1,1}$, and the number of SPY we hold in position is $N_{SPY} = \frac{a}{S_{1,1}}$. The delta of SPY is $\Delta_{SPY} = N_{SPY} = \frac{a}{S_{1,1}}$

By the proof provided in last class, we know that the delta of the option is $\Delta_{Option} = -\frac{1}{S_t}$.

Thus, by delta-hedging, the number of option we hold in position is $N_{option} = |\frac{\Delta_{SPY}}{\Delta_{Option}}| = a$.

Then we do weekly delta-hedge on short of option and do daily delta-hedge on long of the same option. And we do not buy or sell any option after that.

Suppose at the j-th day of i-th week, we have $N_{option} = a$ T option and $N_{option} = -a$ T option in position. we have the exposure $Exposure = N_{option} \times \Delta_{option,daily} - N_{option} \times \Delta_{option,weekly} = a(\frac{1}{S_{i,j}} - \frac{1}{S_{i,1}})$

0.6 Implementation of Strategy

In [121]:
```
df = spy.history(start='2010-01-01', end='2014-12-31', interval='1d')

df = pd.DataFrame(df['Close'])
df['weekly_delta'] = 0
df['daily_delta'] = 0
df['PnL'] = 0

a = 1
```

In [122]:
```
for i, date in zip(range(len(df)), df.index):
    daily_delta = 1 / df.loc[date, 'Close']
    df.loc[date, 'daily_delta'] = daily_delta
    if i%5 == 0:
        weekly_delta = 1 / df.loc[date, 'Close']
        df.loc[date, 'weekly_delta'] = weekly_delta
    else:
        weekly_delta = df.iloc[i-1]['weekly_delta']
        df.loc[date, 'weekly_delta'] = weekly_delta
```

In [123]:
```
for i, date in zip(range(1, len(df)), df.index[1:]):
    profit = a * (df.iloc[i-1]['daily_delta'] - df.iloc[i-1]['weekly_delta']) * (df.
    df.loc[date, 'PnL'] = profit + df.iloc[i-1]['PnL']
```

In [125]:
```
plt.figure(figsize=(10, 6))
_ = plt.plot(df.index, df['PnL'])
_ = plt.xlabel('Time')
_ = plt.ylabel('PnL')
_ = plt.title('Strategy PnL')
```