FINS4792 Problem Set 1

Charlie Bradford
z5114682

January 2020

Chapter 1 Problem Set

1. **Call auction** Graph total market demand and supply curves in price, quantity space for a call auction market where the following orders are submitted to a central auctioneer:

   - Limit orders to buy: 100 shares at \$3.00, 200 shares at \$4.00, 200 shares at \$3.50, and 500 shares at \$2.50.
   - Limit orders to sell: 500 shares at \$5.00, 600 shares at \$3.00, and 500 shares at \$4.00.
   - Market orders to buy: a total of 500 shares.
   - Market orders to sell: a total of 200 shares.

   What is the market clearing price? What quantity of stock is traded? Are all orders that are executable at the market clearing price fully filled?

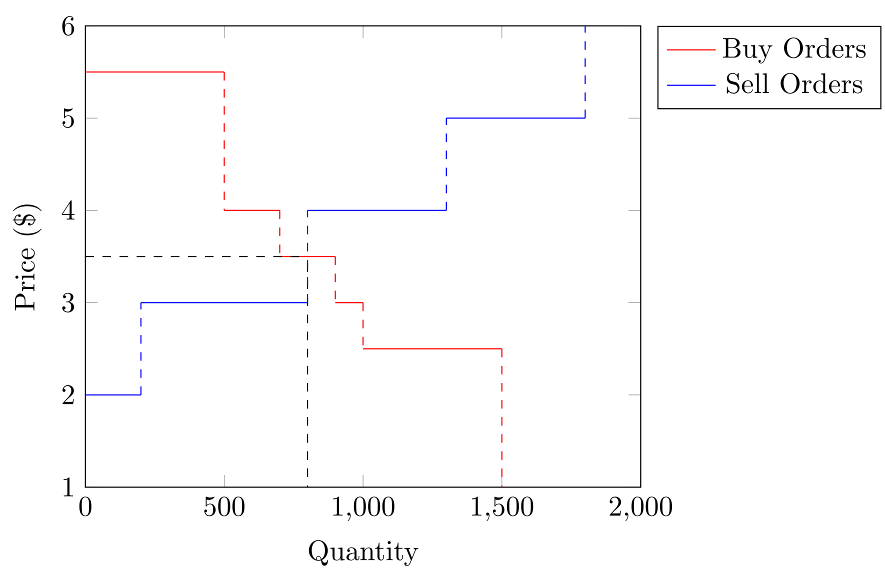


Figure 1: Supply and demand curves for call auction market, showing a clearing price of \$3.50 and a total volume cleared of 800 stocks. The order for 200 shares at \$3.50 is only filled for 100.

---

2. **Continuous order-driven market** Now suppose that the above orders arrive on the market over time, in the order of arrival that is listed above. Track the state of the LOB and the time, price, and quantity of any transactions that take place. Record the dollar bid-ask spread, that is, the difference between the lowest ask and the highest bid, in the continuous market as it evolves from $t = 5$ onwards.

State 1				State 2			
Spread = N/A				Spread = N/A			
100	**3.00**			**200**	**4.00**		
				100	3.00		
State 2				State 3			
Spread = N/A				Spread = N/A			
200	4.00			200	4.00		
200	**3.50**			200	3.50		
100	3.00			100	3.00		
				500	**2.50**		
State 5				State 6			
Spread = 1.00				Spread = 0.50			
200	4.00	**5.00**	**500**	~~200~~ 0	4.00	**3.00**	**~~600~~ 100**
200	3.50			~~200~~ 0	3.50	5.00	500
100	3.00			~~100~~ 0	3.00		
500	2.50			500	2.50		
State 7				State 8			
Spread = 0.50				Spread = 1.50			
500	2.50	3.00	100	**~~500~~ 0**	∞	3.00	~~100~~ 0
		4.00	**500**	2.50	3.00	4.00	~~500~~ 100
		5.00	500			5.00	500
State 9				Final LOB			
Spread = 1.50				Spread = 1.50			
~~500~~ 300	2.50	$-\infty$	**~~200~~ 0**	300	2.50	4.00	100
		4.00	100			5.00	500
		5.00	500			5.00	500

Table 1: Evolution of LOB in a continuous market as orders are placed.

---

3. **Comparison: efficiency and market presence** Consider again the two markets described in questions 1 and 2. Assume that the limit order prices are equal to the order placer's valuation for the block of shares submitted in the order, and think of market orders as placed by agents whose valuation is well outside the relevant range of trading prices. Which market is Pareto efficient, in the sense that at the end of the trading day there is no pair of agents who could both benefit by trading with each other? Intuitively, why?

In the call auction market 800 shares are traded, compared to the 1200 traded in the continuous order driven market. This is because the market (liquidity taking) orders are dealt with first, at prices that are significantly better than their valuations. This reduces the number of total number of fills as the spread is now significantly widened. The order driven market is more efficient, but only due to the specific ordering of the orders, which is not controllable. By modifying the call auction market so that market orders are not applied until an equilibrium price has been found from limit orders alone, a more efficient market could be created.

Chapter 2 Problem Set

1.

$$
\begin{aligned}
s &\equiv \frac{S}{m} \\
&= \frac{a-b}{(a+b)/2} \\
s_q &= \frac{a_q-b_q}{(a_q+b_q)/2} \\
s_{100} &= \frac{74.48-74.42}{(74.48+74.42)/2} \\
&= \frac{0.06}{74.45} \\
&= 0.008054 \\
s_{500} &= \frac{74.48-74.36}{(74.48+74.36)/2} \\
&= \frac{0.12}{74.42} \\
&= 0.01617 \\
s_{1000} &= \frac{76.00-74.36}{(76.00+74.36)/2} \\
&= \frac{1.64}{75.16} \\
&= 0.02182 \\
s_{2000} &= \frac{77.35-73.75}{(77.35-73.75)/2} \\
&= \frac{3.60}{75.55} \\
&= 47.65
\end{aligned}
$$

The bid is deeper for larger orders.

2. (a)

$$\begin{aligned}
S_{9:30} &= 102.31 - 99.50 \\
&= 2.81 \\
s_{9:30} &= \frac{2.81}{(102.31 + 99.50)/2} \\
&= \frac{2.81}{100.905} \\
&= 0.02785 \\
S_{10:30} &= 102.55 - 100.02 \\
&= 2.33 \\
s_{10:30} &= \frac{2.33}{(102.55 + 100.02)/2} \\
&= \frac{2.33}{101.185} \\
&= 0.02303
\end{aligned}$$

(b)

$$\begin{aligned}
S^e &= d(p - m) \\
S^e_{9:30} &= 102.76 - 100.905 \\
&= 1.855 \\
s^e_{9:30} &= \frac{1.855}{100.905} \\
&= 0.01838
\end{aligned}$$

(c) The quoted ask-side half-spread is lower (1.405) than the effective half spread (1.855) due to slippage caused by the large order side.

(d)

$$\begin{aligned}
S_r &= 2 \times d_t(p_t - m_{t+\Delta}) \\
&= 2 \times (102.76 - 101.185) \\
&= 3.15
\end{aligned}$$

(e) The large order has impacted the market.

3. Examining the differences in effects of market orders of size q:

q	Spread
50	0
150	\$1.00
250	\$1.00

4. Roll's model (with all assumptions bar the one stipulated in the question):

$$\begin{aligned}
a_t &= m_t + \frac{S}{2} \\
b_t &= m_t - \frac{S}{2} \\
p_t &= m_t + \frac{S}{2} d_t \\
\Delta p_t &= \frac{S}{2} \Delta d_t \\
S &= 2\sqrt{-cov(\Delta p_{t+1}, \Delta p_t)}
\end{aligned}$$

Now, modifying this model:

$$
\begin{aligned}
E[a_t] &= m_t + \lambda\frac{S}{2} \\
E[b_t] &= m_t - \lambda\frac{S}{2} \\
E[p_t] &= m_t + \lambda\frac{S}{2}d_t \\
E[\Delta p_t] &= \lambda\frac{S}{2}\Delta d_t \\
S &= 2\lambda\sqrt{-cov(\Delta p_{t+1}, \Delta p_t)}
\end{aligned}
$$

FINS4792 Problem Set 2

Charlie Bradford
z5114682

January 2020

1.

$$
\begin{aligned}
s &\equiv \frac{S}{m} \\
&= \frac{a-b}{(a+b)/2} \\
s_q &= \frac{a_q-b_q}{(a_q+b_q)/2} \\
s_{100} &= \frac{74.48-74.42}{(74.48+74.42)/2} \\
&= \frac{0.06}{74.45} \\
&= 0.008054 \\
s_{500} &= \frac{74.48-74.36}{(74.48+74.36)/2} \\
&= \frac{0.12}{74.42} \\
&= 0.01617 \\
s_{1000} &= \frac{76.00-74.36}{(76.00+74.36)/2} \\
&= \frac{1.64}{75.16} \\
&= 0.02182 \\
s_{2000} &= \frac{77.35-73.75}{(77.35-73.75)/2} \\
&= \frac{3.60}{75.55} \\
&= 47.65
\end{aligned}
$$

The bid is deeper for larger orders.

2. (a)

$$\begin{aligned}
S_{9:30} &= 102.31 - 99.50 \\
&= 2.81 \\
s_{9:30} &= \frac{2.81}{(102.31 + 99.50)/2} \\
&= \frac{2.81}{100.905} \\
&= 0.02785 \\
S_{10:30} &= 102.55 - 100.02 \\
&= 2.33 \\
s_{10:30} &= \frac{2.33}{(102.55 + 100.02)/2} \\
&= \frac{2.33}{101.185} \\
&= 0.02303
\end{aligned}$$

(b)

$$\begin{aligned}
S^e &= d(p - m) \\
S^e_{9:30} &= 102.76 - 100.905 \\
&= 1.855 \\
s^e_{9:30} &= \frac{1.855}{100.905} \\
&= 0.01838
\end{aligned}$$

(c) The quoted ask-side half-spread is lower (1.405) than the effective half spread (1.855) due to slippage caused by the large order side.

(d)

$$\begin{aligned}
S_r &= 2 \times d_t(p_t - m_{t+\Delta}) \\
&= 2 \times (102.76 - 101.185) \\
&= 3.15
\end{aligned}$$

(e) The large order has impacted the market.

q	Spread
50	0
150	\$1.00
250	\$1.00

Table 1: Caption

3.

Homework

April 9, 2020

1 Chapter 2

1.1 Question 5

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

data = pd.read_excel("data_2/CH2_AGF_data.xls",
                     header=1,
                     parse_dates=True,
                     dtype={"Time":datetime.time},
                     index_col="Time")
data.head()
```

[2]:
```
                     Trade Size (|qt|)  Price (pt)  Direction (dt)  Bid(bt)  \
Time
2020-04-07 09:06:04                 20     66.7000              -1    66.90
2020-04-07 09:06:11                 25     66.6360              -1    66.65
2020-04-07 09:06:26                 18     66.6000              -1    66.60
2020-04-07 09:07:18                273     66.4163              -1    66.50
2020-04-07 09:07:36                 27     66.5500               1    66.15

                     Ask (at)
Time
2020-04-07 09:06:04     67.00
2020-04-07 09:06:11     66.70
2020-04-07 09:06:26     66.65
2020-04-07 09:07:18     66.55
2020-04-07 09:07:36     66.55
```

1.1.1 Part A

[3]:
```
data["absolute_spread"] = data["Ask (at)"] - data["Bid(bt)"]
data["midpoint"] = (data["Ask (at)"] + data["Bid(bt)"])/2
data["relative_spread"] = data["absolute_spread"] / data["midpoint"]
data["log_spread"] = data["Ask (at)"].apply(np.log) - data["Bid(bt)"].apply(np.
 ↪log)
print("Average absolute spread: {}, \naverage relative spread: {}, \naverage log␣
 ↪spread {}".format(data["absolute_spread"].mean(),
                                                                              ␣
 ↪             data["relative_spread"].mean(),
                                                                              ␣
 ↪             data["log_spread"].mean()))
data.head()
```

```
Average absolute spread: 0.10616570327552977,
average relative spread: 0.0016086984772629698,
average log spread 0.0016086998354762725
```

[3]:
```
                     Trade Size (|qt|)  Price (pt)  Direction (dt)  Bid(bt)  \
Time
2020-04-07 09:06:04                 20     66.7000              -1    66.90
2020-04-07 09:06:11                 25     66.6360              -1    66.65
2020-04-07 09:06:26                 18     66.6000              -1    66.60
2020-04-07 09:07:18                273     66.4163              -1    66.50
2020-04-07 09:07:36                 27     66.5500               1    66.15

                     Ask (at)  absolute_spread  midpoint  relative_spread  \
Time
2020-04-07 09:06:04     67.00             0.10    66.950         0.001494
2020-04-07 09:06:11     66.70             0.05    66.675         0.000750
2020-04-07 09:06:26     66.65             0.05    66.625         0.000750
2020-04-07 09:07:18     66.55             0.05    66.525         0.000752
2020-04-07 09:07:36     66.55             0.40    66.350         0.006029

                     log_spread
Time
2020-04-07 09:06:04    0.001494
2020-04-07 09:06:11    0.000750
2020-04-07 09:06:26    0.000750
2020-04-07 09:07:18    0.000752
2020-04-07 09:07:36    0.006029
```

[4]:
```
for group in data["absolute_spread"].groupby(pd.Grouper(freq='0.5H',␣
 ↪closed='left')):
    print("Average starting at", group[0].time(), ":", sum(group[1])/
 ↪len(group[1]))
```

```
Average starting at 09:00:00 : 0.11363636363636234
Average starting at 09:30:00 : 0.11428571428571456
Average starting at 10:00:00 : 0.10657894736842248
Average starting at 10:30:00 : 0.0874999999999974
Average starting at 11:00:00 : 0.09393939393939205
Average starting at 11:30:00 : 0.10641025641025618
Average starting at 12:00:00 : 0.2052631578947379
Average starting at 12:30:00 : 0.09705882352941127
Average starting at 13:00:00 : 0.08103448275861805
Average starting at 13:30:00 : 0.1280000000000024
Average starting at 14:00:00 : 0.1153846153846132
Average starting at 14:30:00 : 0.0802631578947334
Average starting at 15:00:00 : 0.07586206896552077
Average starting at 15:30:00 : 0.11122448979592132
Average starting at 16:00:00 : 0.12241379310344817
Average starting at 16:30:00 : 0.1118644067796602
Average starting at 17:00:00 : 0.10000000000000019
```

1.1.2 Part B

[5]:
```python
data["absolute_half_spread"] = data["Ask (at)"] - data["midpoint"]
data["relative_half_spread"] = data["absolute_half_spread"] / data["midpoint"]
data["log_half_spread"] = data["Ask (at)"].apply(np.log) - data["midpoint"].
 ↪apply(np.log)

print("Average absolute spread: {}, \naverage relative spread: {}, \naverage log␣
 ↪spread {}".format(data["absolute_half_spread"].mean(),
                                                                             ␣
 ↪               data["relative_half_spread"].mean(),
                                                                             ␣
 ↪               data["log_half_spread"].mean()))

data[["absolute_half_spread", "relative_half_spread", "log_half_spread"]].head()
```

```
Average absolute spread: 0.05308285163776443,
average relative spread: 0.000804349238631478,
average log spread 0.0008038286971465961
```

[5]:
```
                     absolute_half_spread  relative_half_spread  \
Time
2020-04-07 09:06:04                 0.050              0.000747
2020-04-07 09:06:11                 0.025              0.000375
2020-04-07 09:06:26                 0.025              0.000375
2020-04-07 09:07:18                 0.025              0.000376
2020-04-07 09:07:36                 0.200              0.003014
```

```
                     log_half_spread
Time
2020-04-07 09:06:04         0.000747
2020-04-07 09:06:11         0.000375
2020-04-07 09:06:26         0.000375
2020-04-07 09:07:18         0.000376
2020-04-07 09:07:36         0.003010
```

1.1.3 Part C

[6]:
```python
print("Day vwap =", np.mean(data["Trade Size (|qt|)"] * data['Price (pt) '])/
↪data['Trade Size (|qt|)'].mean())
for group in data.groupby(pd.Grouper(freq='3H', closed='right')):
    print("Total VWAP starting at", group[0].time(), ":", np.
↪mean(group[1]["Trade Size (|qt|)"] * group[1]['Price (pt) '])/group[1]['Trade
↪Size (|qt|)'].mean())
    sell = group[1].loc[group[1]["Direction (dt) "] == -1, ["Trade Size (|qt|)",
↪'Price (pt) ']]
    print("Ask VWAP starting at", group[0].time(), ":", np.mean(sell["Trade Size
↪(|qt|)"] * sell['Price (pt) '])/sell['Trade Size (|qt|)'].mean())
    buy = group[1].loc[group[1]["Direction (dt) "] == 1, ["Trade Size (|qt|)",
↪'Price (pt) ']]
    print("Bid VWAP starting at", group[0].time(), ":", np.mean(buy["Trade Size
↪(|qt|)"] * buy['Price (pt) '])/buy['Trade Size (|qt|)'].mean(), "\n")
```

```
Day vwap = 65.9762592589728
Total VWAP starting at 09:00:00 : 66.00037952521801
Ask VWAP starting at 09:00:00 : 65.90238640544808
Bid VWAP starting at 09:00:00 : 66.09206370221254

Total VWAP starting at 12:00:00 : 66.05732484327085
Ask VWAP starting at 12:00:00 : 65.99628722148495
Bid VWAP starting at 12:00:00 : 66.10385337566221

Total VWAP starting at 15:00:00 : 65.93996300616956
Ask VWAP starting at 15:00:00 : 65.85991259218804
Bid VWAP starting at 15:00:00 : 66.08637827551495
```

1.1.4 Part D

[7]:
```python
data["price -1"] = data["Price (pt) "].shift()
data["d_price"] = data["Price (pt) "] - data["price -1"]
data["d_price -1"] = data["d_price"].shift()
```

```
print("Roll's measure = ", 2 * np.sqrt(-1 * np.mean((data["d_price"] -␣
↪data["d_price"].mean()) * data["d_price -1"] - data["d_price -1"].mean())))
```

```
Roll's measure =  0.10688644184158194
```

[8]:
```
data["ln price -1"] = data["Price (pt) "].apply(np.log).shift()
data["ln d_price"] = data["Price (pt) "].apply(np.log) - data["ln price -1"]
data["ln d_price -1"] = data["ln d_price"].shift()
print("Roll's measure = ", 2 * np.sqrt(-1 * np.mean((data["ln d_price"] -␣
↪data["ln d_price"].mean()) * data["ln d_price -1"] - data["ln d_price -1"].
↪mean())))
```

```
Roll's measure =  0.0028872720462376398
```

1.1.5 Part E

[9]:
```
midpoint_change = []
index = []
total_order_flow = data["Trade Size (|qt|)"].sum()
fraction_order_flow = []
for group in data.groupby(pd.Grouper(freq='15T', closed='right')):
    delta = np.round(group[1]["midpoint"][0] - group[1]["midpoint"][-1],␣
↪decimals=2)
    print("Change in midpoint starting at", group[0], ":", delta)
    midpoint_change.append(delta)
    index.append(group[0])
    fraction_order_flow.append((group[1]["Trade Size (|qt|)"] *␣
↪group[1]["Direction (dt) "]).sum() / total_order_flow)
```

```
Change in midpoint starting at 2020-04-07 09:00:00 : 0.6
Change in midpoint starting at 2020-04-07 09:15:00 : 0.2
Change in midpoint starting at 2020-04-07 09:30:00 : 0.15
Change in midpoint starting at 2020-04-07 09:45:00 : -0.17
Change in midpoint starting at 2020-04-07 10:00:00 : 0.08
Change in midpoint starting at 2020-04-07 10:15:00 : 0.42
Change in midpoint starting at 2020-04-07 10:30:00 : 0.03
Change in midpoint starting at 2020-04-07 10:45:00 : 0.1
Change in midpoint starting at 2020-04-07 11:00:00 : -0.02
Change in midpoint starting at 2020-04-07 11:15:00 : -0.72
Change in midpoint starting at 2020-04-07 11:30:00 : -0.22
Change in midpoint starting at 2020-04-07 11:45:00 : 0.42
Change in midpoint starting at 2020-04-07 12:00:00 : 0.22
Change in midpoint starting at 2020-04-07 12:15:00 : -0.43
Change in midpoint starting at 2020-04-07 12:30:00 : 0.12
Change in midpoint starting at 2020-04-07 12:45:00 : 0.25
Change in midpoint starting at 2020-04-07 13:00:00 : 0.15
Change in midpoint starting at 2020-04-07 13:15:00 : 0.23
```

```
Change in midpoint starting at 2020-04-07 13:30:00 : -0.15
Change in midpoint starting at 2020-04-07 13:45:00 : -0.25
Change in midpoint starting at 2020-04-07 14:00:00 : 0.05
Change in midpoint starting at 2020-04-07 14:15:00 : -0.02
Change in midpoint starting at 2020-04-07 14:30:00 : -0.02
Change in midpoint starting at 2020-04-07 14:45:00 : -0.15
Change in midpoint starting at 2020-04-07 15:00:00 : 0.07
Change in midpoint starting at 2020-04-07 15:15:00 : 0.03
Change in midpoint starting at 2020-04-07 15:30:00 : 0.1
Change in midpoint starting at 2020-04-07 15:45:00 : 0.35
Change in midpoint starting at 2020-04-07 16:00:00 : -0.35
Change in midpoint starting at 2020-04-07 16:15:00 : -0.03
Change in midpoint starting at 2020-04-07 16:30:00 : -0.05
Change in midpoint starting at 2020-04-07 16:45:00 : 0.17
Change in midpoint starting at 2020-04-07 17:00:00 : -0.25
Change in midpoint starting at 2020-04-07 17:15:00 : -0.6
```

[10]:
```python
new_data = pd.DataFrame(data=zip(midpoint_change, fraction_order_flow),
                        index=index,
                        columns=["midpoint_change", "order_flow"])
new_data.head()
```

[10]:

	midpoint_change	order_flow
2020-04-07 09:00:00	0.60	-0.001026
2020-04-07 09:15:00	0.20	0.001441
2020-04-07 09:30:00	0.15	-0.006406
2020-04-07 09:45:00	-0.17	0.006327
2020-04-07 10:00:00	0.08	0.006310

[11]:
```python
import statsmodels.formula.api as sm
formula = "midpoint_change ~ order_flow - 1"
reg = sm.ols(formula, new_data).fit()
reg.summary()
```

```
/usr/local/lib/python3.7/site-packages/statsmodels/tools/_testing.py:19:
FutureWarning: pandas.util.testing is deprecated. Use the functions in the
public API at pandas.testing instead.
  import pandas.util.testing as tm
/usr/local/lib/python3.7/site-packages/statsmodels/compat/pandas.py:23:
FutureWarning: The Panel class is removed from pandas. Accessing it from the
top-level namespace will also be removed in the next version
  data_klasses = (pandas.Series, pandas.DataFrame, pandas.Panel)
```

[11]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
=======
```

```
Dep. Variable:         midpoint_change   R-squared (uncentered):
0.198
Model:                             OLS   Adj. R-squared (uncentered):
0.174
Method:                  Least Squares   F-statistic:
8.148
Date:                 Tue, 07 Apr 2020   Prob (F-statistic):
0.00739
Time:                         10:48:44   Log-Likelihood:
-0.79268
No. Observations:                   34   AIC:
3.585
Df Residuals:                       33   BIC:
5.112
Df Model:                            1
Covariance Type:             nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
order_flow    -4.6202      1.619     -2.854      0.007      -7.913      -1.327
==============================================================================
Omnibus:                         1.489   Durbin-Watson:                   1.447
Prob(Omnibus):                   0.475   Jarque-Bera (JB):                0.552
Skew:                           -0.167   Prob(JB):                        0.759
Kurtosis:                        3.528   Cond. No.                         1.00
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

The parameter λ is estimated to be -4.602 with a $t-$stat of -2.854 which is statistically significant. A 1% increase/decrease in relative order flow would effect a 4.602% decrease/increase in the midpoint price.

1.2 Question 6

[12]:
```
data = pd.read_excel("data_2/Ch2_KrispyKreme_raw_data.xls",
                     header=1,
                     parse_dates=True,
                     dtype={"Time":datetime.time},
                     index_col="Time")
data.head()
```

[12]:

```
                     Type  Bid (bt)  Ask (at)  Price (pt)  Bid Size  Ask Size  \
Time
2020-04-07 09:02:25     0      2.66      3.89         NaN     100.0    1000.0
2020-04-07 09:17:40     0      2.65      3.33         NaN     100.0     100.0
2020-04-07 09:17:40     0      2.67      3.54         NaN    3400.0    5000.0
2020-04-07 09:21:21     0      2.66      3.89         NaN     100.0    1000.0
2020-04-07 09:25:01     0      2.67      3.89         NaN    3400.0    1000.0

                     Trade Size (|qt|) Unnamed: 8
Time
2020-04-07 09:02:25                NaN        NaN
2020-04-07 09:17:40                NaN        NaN
2020-04-07 09:17:40                NaN        NaN
2020-04-07 09:21:21                NaN        NaN
2020-04-07 09:25:01                NaN        NaN
```

1.2.1 Part A

[13]:

```python
data["absolute_spread"] = data["Ask (at)"] - data["Bid (bt)"]
data["midpoint"] = (data["Ask (at)"] + data["Bid (bt)"]) / 2
data["relative_spread"] = data["absolute_spread"] / data["midpoint"]
data["log_spread"] = data["Ask (at)"].apply(np.log) - data["Bid (bt)"].apply(np.
 ↪log)
data.head()
```

[13]:

```
                     Type  Bid (bt)  Ask (at)  Price (pt)  Bid Size  Ask Size  \
Time
2020-04-07 09:02:25     0      2.66      3.89         NaN     100.0    1000.0
2020-04-07 09:17:40     0      2.65      3.33         NaN     100.0     100.0
2020-04-07 09:17:40     0      2.67      3.54         NaN    3400.0    5000.0
2020-04-07 09:21:21     0      2.66      3.89         NaN     100.0    1000.0
2020-04-07 09:25:01     0      2.67      3.89         NaN    3400.0    1000.0

                     Trade Size (|qt|) Unnamed: 8  absolute_spread  midpoint  \
Time
2020-04-07 09:02:25                NaN        NaN             1.23     3.275
2020-04-07 09:17:40                NaN        NaN             0.68     2.990
2020-04-07 09:17:40                NaN        NaN             0.87     3.105
2020-04-07 09:21:21                NaN        NaN             1.23     3.275
2020-04-07 09:25:01                NaN        NaN             1.22     3.280

                     relative_spread  log_spread
Time
2020-04-07 09:02:25         0.375573    0.380083
2020-04-07 09:17:40         0.227425    0.228413
2020-04-07 09:17:40         0.280193    0.282048
```

```
2020-04-07 09:21:21         0.375573     0.380083
2020-04-07 09:25:01         0.371951     0.376331
```

[14]:
```
avg_absolute_spread = []
index = []
for group in data.groupby(pd.Grouper(freq="30T")):
    group_data = group[1]
    index.append(group[0])
    avg_absolute_spread.append(group_data["absolute_spread"].mean())

spread_evolution = pd.DataFrame(data=avg_absolute_spread, index=index,␣
 ↪columns=["avg. absolute spread"])
fig = spread_evolution.plot()
```

1.2.2 Part B

[15]:
```
data = data.fillna(method='ffill')
transaction_data = data.loc[data["Type"] == 1, data.columns.values]
transaction_data["most_recent_trans"] = transaction_data["Price (pt)"].shift()
transaction_data["Unnamed: 8"] = np.where(transaction_data["Price (pt)"] >␣
 ↪transaction_data["midpoint"],
                                          1,
                                          np.where(transaction_data["Price␣
 ↪(pt)"] < transaction_data["midpoint"],
                                                   -1,
                                                   np.
 ↪where(transaction_data["Price (pt)"] > transaction_data["most_recent_trans"],
                                                            1,
                                                            -1
                                                           )
                                                  )
                                         )
transaction_data.head()
```

[15]:
```
                     Type  Bid (bt)  Ask (at)  Price (pt)  Bid Size  Ask Size  \
Time
2020-04-07 09:30:02     1      2.81      3.24        2.99     100.0     100.0
2020-04-07 09:30:02     1      2.81      3.24        2.99     100.0     100.0
2020-04-07 09:32:28     1      2.88      3.15        3.03     100.0     100.0
2020-04-07 09:33:35     1      3.01      3.03        3.03    1600.0     700.0
2020-04-07 09:33:38     1      3.01      3.03        3.02     100.0     100.0

                     Trade Size (|qt|)  Unnamed: 8  absolute_spread  midpoint  \
Time
```

```
2020-04-07 09:30:02            500.0         -1             0.43     3.025
2020-04-07 09:30:02            500.0         -1             0.43     3.025
2020-04-07 09:32:28           2400.0          1             0.27     3.015
2020-04-07 09:33:35            100.0          1             0.02     3.020
2020-04-07 09:33:38            100.0          1             0.02     3.020

                     relative_spread  log_spread  most_recent_trans
Time
2020-04-07 09:30:02         0.142149    0.142389                NaN
2020-04-07 09:30:02         0.142149    0.142389               2.99
2020-04-07 09:32:28         0.089552    0.089612               2.99
2020-04-07 09:33:35         0.006623    0.006623               3.03
2020-04-07 09:33:38         0.006623    0.006623               3.03
```

1.3 Question 7

1.3.1 Part A

[16]:
```python
transaction_data["absolute_half_spread"] = transaction_data["absolute_spread"] /␣
 ↪2
transaction_data["relative_half_spread"] = transaction_data["relative_spread"] /␣
 ↪2
transaction_data["log_half_spread"] = transaction_data["Ask (at)"].apply(np.log)␣
 ↪\
                                    - transaction_data["midpoint"].apply(np.log)
data = transaction_data[["Bid (bt)",
                         "Ask (at)",
                         "Price (pt)",
                         "Trade Size (|qt|)",
                         "Unnamed: 8",
                         "absolute_half_spread",
                         "relative_half_spread",
                         "log_half_spread"
                        ]
                       ].rename(columns = {"Bid (bt)": "bid",
                                           "Ask (at)": "ask",
                                           "Price (pt)": "price",
                                           "Trade Size (|qt|)": "size",
                                           "Unnamed: 8": "direction"
                                          }
                               )
data.head()
```

[16]:
```
                      bid   ask  price   size  direction  \
Time
```

```
2020-04-07 09:30:02  2.81  3.24   2.99   500.0         -1
2020-04-07 09:30:02  2.81  3.24   2.99   500.0         -1
2020-04-07 09:32:28  2.88  3.15   3.03  2400.0          1
2020-04-07 09:33:35  3.01  3.03   3.03   100.0          1
2020-04-07 09:33:38  3.01  3.03   3.02   100.0          1

                     absolute_half_spread  relative_half_spread  \
Time
2020-04-07 09:30:02                 0.215              0.071074
2020-04-07 09:30:02                 0.215              0.071074
2020-04-07 09:32:28                 0.135              0.044776
2020-04-07 09:33:35                 0.010              0.003311
2020-04-07 09:33:38                 0.010              0.003311

                     log_half_spread
Time
2020-04-07 09:30:02         0.068662
2020-04-07 09:30:02         0.068662
2020-04-07 09:32:28         0.043803
2020-04-07 09:33:35         0.003306
2020-04-07 09:33:38         0.003306
```

1.3.2 Part B

[17]:
```python
vwap = (data["price"] * data["size"]).sum() / data["size"].sum()

data["buy_size"] = np.where(data["direction"] == 1, data["size"], 0)
data["sell_size"] = np.where(data["direction"] == -1, data["size"], 0)

vwap_b = (data["price"] * data["buy_size"]).sum() / data["buy_size"].sum()
vwap_s = (data["price"] * data["sell_size"]).sum() / data["sell_size"].sum()

print(f"VWAP: {vwap:.4f}, VWAP Buy: {vwap_b:.4f}, VWAP Sell: {vwap_s:.4f}")
```

```
VWAP: 3.0534, VWAP Buy: 3.0647, VWAP Sell: 3.0454
```

[18]:
```python
for group in data.groupby(pd.Grouper(freq="3H", closed="left")):
    local = group[1].copy()
    local["buy_size"] = np.where(local["direction"] == 1, local["size"], 0)
    local["sell_size"] = np.where(local["direction"] == -1, local["size"], 0)

    vwap = (local["price"] * local["size"]).sum() / local["size"].sum()

    vwap_b = (local["price"] * local["buy_size"]).sum() / local["buy_size"].sum()
    vwap_s = (local["price"] * local["sell_size"]).sum() / local["sell_size"].
 ↪sum()
```

```
    print(f"Time: {group[0]}, VWAP: {vwap:.4f}, VWAP Buy: {vwap_b:.4f}, VWAP␣
↪Sell: {vwap_s:.4f}")
```

```
Time: 2020-04-07 09:00:00, VWAP: 3.0180, VWAP Buy: 3.0357, VWAP Sell: 3.0036
Time: 2020-04-07 12:00:00, VWAP: 3.0868, VWAP Buy: 3.0932, VWAP Sell: 3.0812
Time: 2020-04-07 15:00:00, VWAP: 3.0689, VWAP Buy: 3.0801, VWAP Sell: 3.0649
```

1.3.3 Part C

[23]:
```
data["lagged_price"] = data["price"].shift()
rolls_estimate = 2 * np.sqrt(data["price"].cov(data["lagged_price"]))

data["log_price"] = data["price"].apply(np.log)
data["lagged_log_price"] = data["log_price"].shift()
rolls_estimate_log = 2 * np.sqrt(data["log_price"].cov(data["lagged_log_price"]))

print(rolls_estimate, rolls_estimate_log)
```

```
0.09289507177090663 0.03060879601340561
```

[37]:
```
lag_log_p = []
log_p = []
lag_p = []
p = []
for group in data.groupby(pd.Grouper(freq="15T", closed="left")):
    lag_log_p.append(group[1]["lagged_log_price"][-1])
    log_p.append(group[1]["log_price"][-1])
    lag_p.append(group[1]["lagged_price"][-1])
    p.append(group[1]["price"][-1])


rolls_estimate_log_15 = 2 * np.sqrt(np.cov((lag_log_p, log_p))[0][1])
rolls_estimate_15 = 2* np.sqrt(np.cov((lag_p, p))[0][1])

print(rolls_estimate_15, rolls_estimate_log_15)
```

```
0.09230911280958287 0.030409739390525905
```

1.3.4 Part D

[55]:
```
data["midprice"] = (data["ask"] + data["bid"]) / 2

midprices = []
cumm_flow = []
index = []
```

```python
mid = data["midprice"][0]
for group in data.groupby(pd.Grouper(freq="15T", closed="left")):
    midprices.append(round(group[1]["midprice"][-1] - mid, 4))
    cumm_flow.append(round((group[1]["price"] * group[1]["direction"]).sum(), 4))
    mid = group[1]["midprice"][-1]
    index.append(group[0])



new_data = pd.DataFrame(data=zip(midprices, cumm_flow),
                        index=index,
                        columns=["midpoint_change", "cumm_flow"])

formula = "midpoint_change ~ cumm_flow - 1"
reg = sm.ols(formula, new_data).fit()
reg.summary()
```

[55]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:        midpoint_change   R-squared (uncentered):
0.069
Model:                            OLS   Adj. R-squared (uncentered):
0.032
Method:                 Least Squares   F-statistic:
1.857
Date:                Tue, 07 Apr 2020   Prob (F-statistic):
0.185
Time:                        11:19:05   Log-Likelihood:
19.952
No. Observations:                  26   AIC:
-37.90
Df Residuals:                      25   BIC:
-36.65
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
cumm_flow      0.0004      0.000      1.363      0.185      -0.000       0.001
==============================================================================
Omnibus:                       17.527   Durbin-Watson:                   2.948
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               87.343
Skew:                          -0.384   Prob(JB):                     1.08e-19
```

```
Kurtosis:                          11.946   Cond. No.                         1.00
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

1.4 Question 8

1.4.1 Part A

The optimal return would be $q(m_t - m_0)$. As we must choose some proportion k of the order to purchase, and purchase at an average price of $\bar{p} = m_o + \lambda kq$, the actual return is $kq(m_t - \bar{p})$. This gives a shortfall of:

$$\begin{aligned}
\text{Shortfall} &= q(m_t - m_0) - kq(m_t - \bar{p}) \\
&= qm_t - qm_0 - kqm_t + kq\bar{p} \\
&= qm_t - qm_0 - kqm_t + kq(m_0 + \lambda kq) \\
&= qm_t - qm_0 - kqm_t + kqm_0 + \lambda k^2 q^2 \\
&= q(1-k)(m_t - m_0) + \lambda k^2 q^2
\end{aligned}$$

1.4.2 Part B

$$\begin{aligned}
\frac{d\text{Shortfall}}{dk} &\equiv 0 \\
&= -qm_t + qm_0 + 2\lambda kq^2 \\
k^* &= \frac{m_t - m_0}{2\lambda q}
\end{aligned}$$

1.4.3 Part C

$$\begin{aligned}
\frac{\partial k^*}{\partial(m_t - m_0)} &> 0 \\
\frac{\partial k^*}{\partial q} &< 0 \\
\frac{\partial k^*}{\partial \lambda} &< 0
\end{aligned}$$

2 Chapter 3

2.1 Question 1

2.1.1 Part A

$$
\begin{aligned}
a_t &= \mu_{t-1} + \frac{k}{\bar{q}} + s_t^a \\
b_t &= \mu_{t-1} - \frac{k}{\bar{q}} - s_t^b \\
S_t &\equiv 2\frac{k}{\bar{q}} + s_t^a + s_t^b
\end{aligned}
$$

Where $\bar{q}$ is the average order size.

2.2 Part B

As above:

$$
\begin{aligned}
a_t &= \mu_{t-1} + \frac{kp}{\bar{q}} + s_t^a \\
b_t &= \mu_{t-1} - \frac{kp}{\bar{q}} - s_t^b \\
S_t &\equiv 2\frac{kp}{\bar{q}} + s_t^a + s_t^b
\end{aligned}
$$

2.2.1 Part C

Within this framework, the observations would imply that order-processing costs are constant per dollar of value traded.

2.3 Question 2

2.3.1 Part A

Perfectly competitive dealers with no orderprocessing costs will post bid and ask offers at $\mu = 15$

2.3.2 Part B

In order to still make 0 profit with 10% informed traders, the spread must be widened by $0.10(v^H - v^L) = 1$, giving:

$$
\begin{aligned}
a &= 15 + 0.5 \\
&= 15.5 \\
b &= 15 - 0.5 \\
&= 14.5
\end{aligned}
$$

2.3.3 Part C

An uninformed trader will lose, on average \$0.50. An informed trader will expect to make \$4.50.

2.3.4 Part D

In order to account for insider traders, market makers have to increase their spread, money which is effectively taken from uninformed traders.

2.4 Question 3

Consider the one-period Glosten-Milgrom model, where the security's true value v can be high (vH) or low (vL) with probability 0.5 each. Market makers are competitive and risk neutral, and do not know v. In each period, a single trader comes to the market: with probability 1 , he is a noise trader, who buys or sells one unit with probability 0.5 each; with probability he is an informed trader, who observes a signal about security's true value. With probability (2 , 1] the signal is accurate, that is, it coincides with the true value of the security. With probability 1 , instead, the signal is mistaken, so that the insider assigns the wrong value to the security. Hence, measures the accuracy of the signal observed by the informed trader: for close to 1 , the informed would be similar to a noise trader; for = 1, the insider trader would be perfectly informed.

c. Derive the bid-ask spread as a function of signal's informativeness. How does this result compare with the case of perfectly informed insider trading? Is the market more or less illiquid? Intuitively, why?

d. Verify whether, given the bid and ask prices derived at point b, the insider is actually willing to buy when his signal equals vH and sells when it is vL, that is, whether this strategy yields positive expected profits in equilibrium.

2.4.1 Part A

The dealers expected profit trading with an uniformed trader, and informed trader:

$$\begin{aligned}
p_u &= \frac{a-b}{2} \\
p_i &= (1-\rho)\frac{a-b}{2} + \rho(v^H - a) \\
p &= \pi p_i + (1-\pi)p_u \\
&= \pi(1-\rho)\frac{a-b}{2} + \pi\rho(v^H - a) + (1-\pi)\frac{a-b}{2} \\
&= (1-\pi+\pi(1-\rho))\frac{a-b}{2} + \pi\rho(v^H - a) \\
&= (1-\pi\rho)\frac{a-b}{2} + \pi\rho(v^H - a)
\end{aligned}$$

2.4.2 Part B

$$\begin{aligned}
0 &= (1-\pi\rho)(a-\mu) + \pi\rho(v^H - a) \\
&= (1-\pi\rho)a - (1-\pi\rho)\mu + \pi\rho v^H - \pi\rho a \\
&= (1-2\pi\rho)a - (1-\pi\rho)\mu + \pi\rho v^H \\
a &= \frac{(1-\pi\rho)\mu - \pi\rho v^H}{1-2\pi\rho} \\
&= \frac{(1-\pi\rho)\mu - \pi\rho(\mu + \frac{v^H - v^L}{2})}{1-2\pi\rho} \\
&= \frac{(1-\pi\rho)\mu - \pi\rho\mu + \pi\rho(\frac{v^H - v^L}{2})}{1-2\pi\rho} \\
&= \mu + \frac{\pi\rho(\frac{v^H - v^L}{2})}{1-2\pi\rho} \\
b &= \mu - \frac{\pi\rho(\frac{v^H - v^L}{2})}{1-2\pi\rho}
\end{aligned}$$

2.4.3 Part C

$$\begin{aligned}
S &= 2\frac{\pi\rho(\frac{v^H - v^L}{2})}{1-2\pi\rho} && (1) \\
&= \frac{\pi\rho(v^H - v^L)}{1-2\pi\rho} && (2) \\
\frac{\partial S}{\partial \rho} &= \frac{(1-2\pi\rho)(\pi(v^H - v^L)) - (\pi\rho(v^H - v^L))(1-2\pi)}{(1-2\pi\rho)^2} && (3) \\
&= \frac{\pi(1-\rho)(v^H - v^L)}{(1-2\pi\rho)^2} && (4)
\end{aligned}$$

2.4.4 Part D

The expected profit of the trader is

$$\begin{aligned}
p &= \rho(\frac{v^H - v^L}{2} - \frac{a-b}{2}) - (1-\rho)(\frac{a-b}{2}) && (5) \\
&= \rho\frac{v^H - v^L}{2} - \rho\frac{a-b}{2} - \frac{a-b}{2} + \rho\frac{a-b}{2} && (6) \\
&= \rho\frac{v^H - v^L}{2} - \frac{a-b}{2} && (7) \\
0 &\leq \rho\frac{v^H - v^L}{2} - \frac{a-b}{2} && (8) \\
a - b &\leq \rho(v^H - v^L) && (9)
\end{aligned}$$

The strategy makes a profit if ρ times the difference between the high and low values is greater than the spread.

2.5 Question 4

2.5.1 Part A

In this market a there is a $(1-\pi)+\phi\pi$ chance that a participant will traded. The profit made of noise traders is $\frac{a-b}{2}$ and the loss incurred from informed traders is $\frac{v^H - v^L}{2} - \frac{a-b}{2}$. This yields a total profit by market makers of:

$$p = (1-\pi)\frac{a-b}{2} - \phi\pi(\frac{v^H - v^L}{2} - \frac{a-b}{2})$$

In a perfectly competitive market, this equals 0, giving:

$$\begin{aligned}
0 &= (1-\pi)\frac{a-b}{2} - \phi\pi(\frac{v^H - v^L}{2} - \frac{a-b}{2}) && (10)\\
&= (1-\pi)(a-\mu) - \phi\pi(\frac{v^H - v^L}{2} - a + \mu) && (11)\\
&= (1-\pi)a - (1-\pi)\mu - \phi\pi\frac{v^H - v^L}{2} - \phi\pi a + \phi\pi\mu && (12)\\
&= (1-\pi-\phi\pi)a - (1-\pi-\phi\pi)\mu - \phi\pi\frac{v^H - v^L}{2} && (13)\\
(1-\pi-\phi\pi)a &= (1-\pi-\phi\pi)\mu + \phi\pi\frac{v^H - v^L}{2} && (14)\\
a &= \frac{(1-\pi-\phi\pi)\mu + \phi\pi\frac{v^H - v^L}{2}}{1-\pi-\phi\pi} && (15)\\
&= \mu + \frac{\phi\pi\frac{v^H - v^L}{2}}{1-\pi-\phi\pi} && (16)\\
b &= \mu - \frac{\phi\pi\frac{v^H - v^L}{2}}{1-\pi-\phi\pi} && (17)
\end{aligned}$$

2.5.2 Part B

$$\begin{aligned}
p &= \frac{v^H - v^L}{2} - \frac{\phi\pi\frac{v^H - v^L}{2}}{1-\pi-\phi\pi} - c && (18)\\
&= \left(1 - \frac{\phi\pi}{1-\pi-\phi\pi}\right)\frac{v^H - v^L}{2} - c && (19)
\end{aligned}$$

2.5.3 Part C

$$\begin{aligned}
\phi &= 0 && (20)\\
0 &\leq \left(1 - \frac{0}{1-\pi-0}\right)\frac{v^H - v^L}{2} - c && (21)\\
c &\leq \frac{v^H - v^L}{2} && (22)
\end{aligned}$$

2.5.4 Part D

$$\phi = 1 \tag{23}$$
$$0 \leq \left(1 - \frac{\pi}{1 - \pi - \pi}\right) \frac{v^H - v^L}{2} - c \tag{24}$$
$$c \leq \left(1 - \frac{\pi}{1 - 2\pi}\right) \frac{v^H - v^L}{2} \tag{25}$$

2.5.5 Part E

The traders will be indifferent about trading when the expected profit is zero. I.e.

$$c = \left(1 - \frac{\phi\pi}{1 - \pi - \phi\pi}\right) \frac{v^H - v^L}{2}$$

It obviously depends quite significantly on the π, the assumed proportion of informed traders, and the potential profit, $\frac{v^H - v^L}{2}$. Determing ϕ based on c:

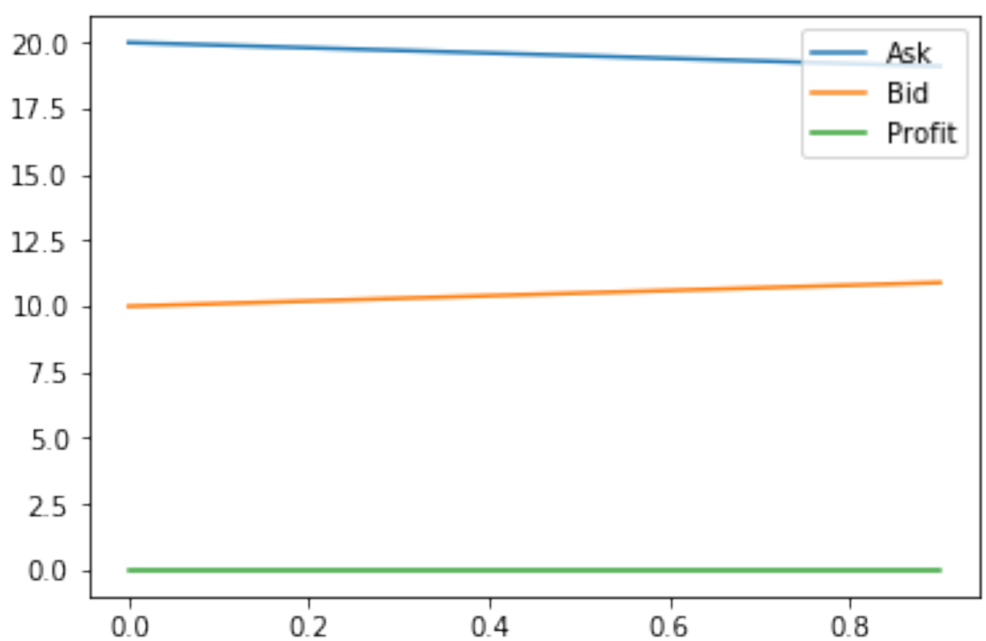
$$c = \left(1 - \frac{\phi\pi}{1 - \pi - \phi\pi}\right) \frac{v^H - v^L}{2} \tag{26}$$
$$= \frac{v^H - v^L}{2} - \frac{\phi\pi}{1 - \pi - \phi\pi} \frac{v^H - v^L}{2} \tag{27}$$
$$(1 - \pi - \phi\pi)c = (1 - \pi - \phi\pi)\frac{v^H - v^L}{2} - \phi\pi\frac{v^H - v^L}{2} \tag{28}$$
$$(1 - \pi)c - (1 - \pi)\frac{v^H - v^L}{2} = \pi\phi c - 2\pi\phi\frac{v^H - v^L}{2} \tag{29}$$
$$\phi = \frac{(1 - \pi)c - (1 - \pi)\frac{v^H - v^L}{2}}{\pi c - 2\pi\frac{v^H - v^L}{2}} \tag{30}$$

2.5.6 Part F

[24]:
```
pi = 0.9
v_H = 20
v_L = 10

phi_f      = lambda c : ((1 - pi) * c - (1 - pi) * ((v_H - v_L) / 2)) / (pi * c␣
 ↪- 2 * pi * ((v_H - v_L) / 2))
h_spread_f = lambda c : (phi_f(c) * pi * ((v_H - v_L) / 2)) / (1 - pi - phi_f(c)␣
 ↪* pi)
ask_f      = lambda c : (v_H + v_L) / 2 + h_spread_f(c)
bid_f      = lambda c : (v_H + v_L) / 2 - h_spread_f(c)
profit_f   = lambda c : (v_H - v_L) / 2 - h_spread_f(c) - c

n_p    = 10
```

```
c      = [x / n_p for x in range(n_p)]
ask    = [ask_f(x) for x in c]
bid    = [bid_f(x) for x in c]
profit = [profit_f(x) for x in c]

import matplotlib.pyplot as ppt
print("c, Bid,   Ask,    Profit")
for x, a, b, p in zip(c, ask, bid, profit):
    print(f"{x:3.3f}, {b:5.2f}, {a:5.2f}, {p:5.2f}")
ppt.plot(c, list(zip(ask, bid, profit)))
ppt.legend(["Ask", "Bid", "Profit"])
ppt.show()
```

```
c, Bid,   Ask,    Profit
0.000, 10.00, 20.00,  0.00
0.100, 10.10, 19.90,  0.00
0.200, 10.20, 19.80,  0.00
0.300, 10.30, 19.70, -0.00
0.400, 10.40, 19.60, -0.00
0.500, 10.50, 19.50,  0.00
0.600, 10.60, 19.40,  0.00
0.700, 10.70, 19.30, -0.00
0.800, 10.80, 19.20, -0.00
0.900, 10.90, 19.10, -0.00
```



2.6 Question 5

2.7 Part A

$$0 = (1-\pi)\begin{cases} \frac{a-b}{2} & \frac{a-b}{2} \leq \delta \\ 0 \end{cases} - \pi\left(\frac{v^H - v^L}{2} - \frac{a-b}{2}\right) \tag{31}$$

$$\pi\left(\frac{v^H - v^L}{2} - a + \mu\right) = (1-\pi)\begin{cases} a-\mu & a-\mu \leq \delta \\ 0 \end{cases} \tag{32}$$

$$\pi\left(\frac{v^H - v^L}{2}\right) - \pi a + \pi\mu = \begin{cases} (1-\pi)a - (1-\pi)\mu & a-\mu \leq \delta \\ 0 \end{cases} \tag{33}$$

$$\pi\left(\frac{v^H - v^L}{2}\right) = \begin{cases} a-\mu & a-\mu \leq \delta \\ \pi a - \pi\mu \end{cases} \tag{34}$$

$$\pi\left(\frac{v^H - v^L}{2}\right) = (a-\mu) * \begin{cases} 1 & a-\mu \leq \delta \\ \pi \end{cases} \tag{35}$$

$$\pi\left(\frac{v^H - v^L}{2}\right) = a - \mu - (a-\mu) * \begin{cases} 0 & a-\mu \leq \delta \\ 1-\pi \end{cases} \tag{36}$$

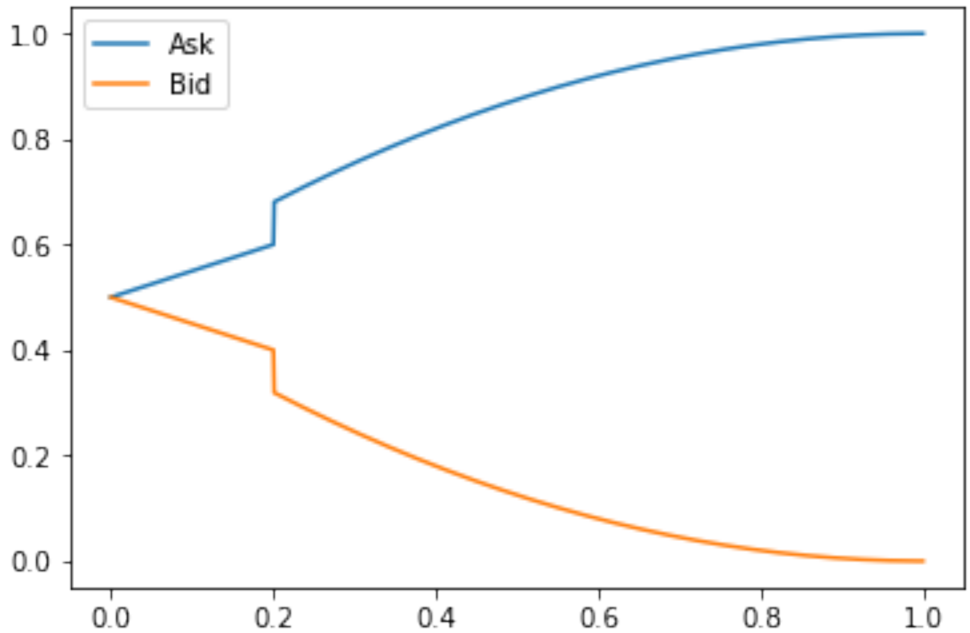
$$a = \mu + \pi\left(\frac{v^H - v^L}{2}\right) + (a-\mu) * \begin{cases} 0 & a-\mu \leq \delta \\ 1-\pi \end{cases} \tag{37}$$

$$b = \mu - \pi\left(\frac{v^H - v^L}{2}\right) - (b-\mu) * \begin{cases} 0 & b-\mu \leq \delta \\ 1-\pi \end{cases} \tag{38}$$

2.7.1 Part B

[26]:
```
v_H = 1
v_L = 0
mu  = 0.5
delta = 0.1

h_spread_f = lambda pi : pi * (v_H - v_L) / 2
ask_f      = lambda pi: mu + h_spread_f(pi) + ((1 - pi) * h_spread_f(pi) if␣
 ↪h_spread_f(pi) > delta else 0)
bid_f      = lambda pi: mu - h_spread_f(pi) - ((1 - pi) * h_spread_f(pi) if␣
 ↪h_spread_f(pi) > delta else 0)

np   = 1000
pi_l = [x / np for x in range(np)]
ask  = [ask_f(x) for x in pi_l]
bid  = [bid_f(x) for x in pi_l]

ppt.plot(pi_l, list(zip(ask, bid)))
ppt.legend(["Ask", "Bid"])
```

```
ppt.show()
```



2.8 Question 6

Expected profit from trading with a noise trader:

$$p_u = \frac{a-b}{2}$$

Expected profit from trading with an informed trader:

$$p_i = \frac{a-b}{2} - \begin{cases} 2a+1 & v \geq a \\ 0 & b < v < a \\ \frac{b}{b+2} & v \leq b \end{cases}$$

Given perfectly competitive markets:

$$0 = (1-\pi)\frac{a-b}{2} + \pi \left(\frac{a-b}{2} - \begin{cases} 2a+1 & v \geq a \\ 0 & b < v < a \\ \frac{b}{b+2} & v \leq b \end{cases} \right) \tag{39}$$

$$= (1-\pi)\frac{a-b}{2} + \pi\frac{a-b}{2} - \pi \begin{cases} 2a+1 & v \geq a \\ 0 & b < v < a \\ \frac{b}{b+2} & v \leq b \end{cases} \tag{40}$$

$$= \frac{a-b}{2} - \pi \begin{cases} 2a+1 & v \geq a \\ 0 & b < v < a \\ \frac{b}{b+2} & v \leq b \end{cases} \tag{41}$$

$$= a - \mu - \pi \begin{cases} 2a+1 & v \geq a \\ 0 & v < a \end{cases} \tag{42}$$

$$a = \mu + \pi \begin{cases} 2a+1 & v \geq a \\ 0 & v < a \end{cases} \tag{43}$$

$$= \mu + F(a)\pi(2a+1) \tag{44}$$

$$b = \mu - (1-F(b))\pi\frac{b}{b+2} \tag{45}$$

[64]:

```
F = lambda x : (1 - (1 / (x + 1) ** 2))




def get_a(pi):
    ask_f = lambda x : 1 + F(x) * pi * (2 * a + 1)
    bid_f = lambda x : 1 - (1-F(x)) * pi * (x / (x + 1))
    a = 1
    e = 1e-20
    l = 0.1

    while abs(a-ask_f(a)) > e:
        if a > ask_f(a):
            a -= l
        else:
            a += l
        l = l - l/100
        if l < e:
            return float("inf")
    return a

def get_b(pi):
    ask_f = lambda x : 1 + F(x) * pi * (2 * a + 1)
```

```
    bid_f = lambda x : 1 - (1-F(x)) * pi * (x / (x + 1))
    b = 1
    e = 1e-20
    l = 0.1

    while abs(b-bid_f(b)) > e:
        if b > bid_f(b):
            b -= l
        else:
            b += l
        l = l - l/100
        if l < e:
            return float("-inf")
    return b
np = 100_000
prev = 1
print("Critical pi")
for x in range(int(np * 0.434), np):
    curr = get_a(x/np)
    if curr == float("inf"):
        print(prev)
        break
    prev = x/np
    # print(f"{x/np:5.3f}, {get_a(x/np):5.3f}, {get_b(x/np):5.3f}")
```

```
Critical pi
0.43782
```

2.9 Question 7

The price pressure of inventory holdings scales with the illiquidity of the holdings. Thus small cap, illiquid stocks produce greater price pressure than large cap, liquid stocks.

2.10 Question 8

2.10.1 Part B

$$p = \kappa_l(1-\pi)\frac{a-b}{2} - \kappa_i\pi(0.5 - \frac{a-b}{2}) \tag{46}$$
$$0 = \kappa_l(1-\pi)(a-0.5) - \kappa_i\pi(0.5 - a + 0.5) \tag{47}$$
$$= \kappa_l(1-\pi)a - \kappa_l(1-\pi)*0.5 - \kappa_i\pi + \kappa_i\pi a \tag{48}$$
$$= (\kappa_l(1-\pi) + \kappa_i\pi)a - \kappa_l(1-\pi)*0.5 - \kappa_i\pi \tag{49}$$
$$a = \frac{\kappa_l(1-\pi)*0.5 + \kappa_i\pi}{\kappa_l(1-\pi) + \kappa_i\pi} \tag{50}$$
$$= \frac{\kappa(1-\pi)*0.5 + \kappa\pi}{\kappa(1-\pi) + \kappa\pi} \tag{51}$$
$$= (1-\pi)*0.5 + \pi \tag{52}$$
$$= \mu + \pi\frac{v^H - v^L}{2} \tag{53}$$
$$b = \mu - \pi\frac{v^H - v^L}{2} \tag{54}$$

If k_i increases then the spread decreases, else it increases. This makes sense as fewer informed traders lead to a narrow spread.

2.11 Question 12

[1]:
```
import pandas as pd
import numpy as np
data = pd.read_excel("Ch3_ex12_data.xls")
data.head()
```

[1]:
```
  ticker          pb          pa           p            vo            vp  \
0   AACC    6.376667    6.400000    6.393333     60.100000  3.831219e+02
1   AAME    0.500333    0.644567    0.576556      5.333333  2.940111e+00
2   AAPL  116.898750  116.975000  116.970000  18880.875000  2.199989e+06
3   AATI    4.091111    4.111111    4.108889    263.977778  1.105406e+03
4   AAWW   20.101111   20.142222   20.133333    282.211111  5.661318e+03

   ibnosh  gics     bas100    ami100         mktcap       ret      vola
0  30.573    40   0.368915  0.138675     195.463378  0.012650  0.058134
1  21.903    40  24.404827  2.314466      12.628297  0.038419  0.223454
2 890.554    20   0.065185  0.000100  104168.087500  0.007268  0.028199
3  42.891    20   0.491540  0.017150     176.234344  0.022492  0.036442
4  21.062    20   0.203990  0.020136     424.048278  0.022644  0.062385
```

2.11.1 Part A

[11]:
```
data["spread"] = data["pa"] - data["pb"]
x = np.corrcoef(data[["p", "vp", "mktcap", "spread", "ret", "vola"]],␣
↪rowvar=False)
for row in x:
    for v in row:
        print(f"{v:+6.4f} ", end="")
    print()
```

```
+1.0000 +0.4999 +0.3718 +0.1170 -0.0641 -0.2537
+0.4999 +1.0000 +0.8923 -0.0150 -0.0138 -0.0815
+0.3718 +0.8923 +1.0000 -0.0202 -0.0275 -0.0966
+0.1170 -0.0150 -0.0202 +1.0000 -0.0416 +0.0289
-0.0641 -0.0138 -0.0275 -0.0416 +1.0000 +0.3311
-0.2537 -0.0815 -0.0966 +0.0289 +0.3311 +1.0000
```

The most correlated values are market cap and trading volume. Be careful not to induce omitted variable bias.

2.11.2 Part B

[24]:
```
data["turnover"] = data["vo"] / data["ibnosh"]
data["l_vola"] = data["vola"].apply(np.log)

formulas = ["spread ~ vola + " for _ in range(6)]
formulas[0] += "mktcap"
formulas[1] += "vp"
formulas[2] += "np.log(mktcap)"
formulas[3] += "l_vola"
formulas[4] += "turnover"
formulas[5] += "turnover + np.log(p)"

import statsmodels.formula.api as sm
```

[25]:
```
sm.ols(formulas[0], data).fit().summary()
```

[25]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                 -0.000
Method:                 Least Squares   F-statistic:                    0.8616
Date:                Mon, 13 Apr 2020   Prob (F-statistic):              0.423
Time:                        14:14:32   Log-Likelihood:                 527.13
No. Observations:                1514   AIC:                            -1048.
```

```
Df Residuals:                    1511   BIC:                            -1032.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0707      0.009      8.262      0.000       0.054       0.088
vola           0.1540      0.146      1.053      0.293      -0.133       0.441
mktcap     -3.882e-07   5.72e-07     -0.679      0.497   -1.51e-06    7.33e-07
==============================================================================
Omnibus:                     1906.659   Durbin-Watson:                   1.960
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           241272.541
Skew:                           6.759   Prob(JB):                         0.00
Kurtosis:                      63.348   Cond. No.                     2.61e+05
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.61e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[26]:
```python
sm.ols(formulas[1], data).fit().summary()
```

[26]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                 -0.000
Method:                 Least Squares   F-statistic:                    0.7522
Date:                Mon, 13 Apr 2020   Prob (F-statistic):              0.472
Time:                        14:14:33   Log-Likelihood:                 527.02
No. Observations:                1514   AIC:                            -1048.
Df Residuals:                    1511   BIC:                            -1032.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0705      0.009      8.234      0.000       0.054       0.087
vola           0.1578      0.146      1.079      0.281      -0.129       0.444
vp         -2.288e-08   4.65e-08     -0.492      0.622   -1.14e-07    6.82e-08
==============================================================================
Omnibus:                     1906.601   Durbin-Watson:                   1.960
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           241216.870
```

```
Skew:                           6.759   Prob(JB):                         0.00
Kurtosis:                      63.341   Cond. No.                     3.21e+06
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 3.21e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[27]:
```
sm.ols(formulas[2], data).fit().summary()
```

[27]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.037
Model:                            OLS   Adj. R-squared:                  0.036
Method:                 Least Squares   F-statistic:                     29.35
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           3.13e-13
Time:                        14:14:33   Log-Likelihood:                 555.11
No. Observations:                1514   AIC:                            -1104.
Df Residuals:                    1511   BIC:                            -1088.
Df Model:                           2
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
Intercept          0.1972      0.019     10.516      0.000       0.160
0.234
vola              -0.2471      0.153     -1.616      0.106      -0.547
0.053
np.log(mktcap)    -0.0206      0.003     -7.575      0.000      -0.026
-0.015
==============================================================================
Omnibus:                     1911.255   Durbin-Watson:                   1.948
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           248788.377
Skew:                           6.776   Prob(JB):                         0.00
Kurtosis:                      64.320   Cond. No.                         197.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
```

```
specified.
"""
```

[32]:
```
sm.ols(formulas[3], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
  ↪summary()
```

[32]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.015
Model:                            OLS   Adj. R-squared:                  0.013
Method:                 Least Squares   F-statistic:                     11.17
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           1.53e-05
Time:                        14:16:46   Log-Likelihood:                 536.03
No. Observations:                1512   AIC:                            -1066.
Df Residuals:                    1509   BIC:                            -1050.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -0.2145      0.063     -3.428      0.001      -0.337      -0.092
vola           1.3561      0.297      4.570      0.000       0.774       1.938
l_vola        -0.0713      0.016     -4.576      0.000      -0.102      -0.041
==============================================================================
Omnibus:                     1922.264   Durbin-Watson:                   1.969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           254553.921
Skew:                           6.859   Prob(JB):                         0.00
Kurtosis:                      65.067   Cond. No.                         233.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[33]:
```
sm.ols(formulas[4], data).fit().summary()
```

[33]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.013
Model:                            OLS   Adj. R-squared:                  0.012
Method:                 Least Squares   F-statistic:                     10.17
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           4.12e-05
```

```
Time:                        14:17:01   Log-Likelihood:                 536.38
No. Observations:                1514   AIC:                            -1067.
Df Residuals:                    1511   BIC:                            -1051.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0777      0.009      9.054      0.000       0.061       0.095
vola           0.1832      0.145      1.264      0.206      -0.101       0.467
turnover      -0.0011      0.000     -4.365      0.000      -0.002      -0.001
==============================================================================
Omnibus:                     1910.272   Durbin-Watson:                   1.959
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           244110.763
Skew:                           6.778   Prob(JB):                         0.00
Kurtosis:                      63.711   Cond. No.                         642.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[34]:

```
sm.ols(formulas[5], data).fit().summary()
```

[34]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.042
Model:                            OLS   Adj. R-squared:                  0.040
Method:                 Least Squares   F-statistic:                     22.16
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           4.80e-14
Time:                        14:17:05   Log-Likelihood:                 558.89
No. Observations:                1514   AIC:                            -1110.
Df Residuals:                    1510   BIC:                            -1088.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0103      0.013      0.787      0.431      -0.015       0.036
vola           0.6449      0.158      4.074      0.000       0.334       0.955
turnover      -0.0014      0.000     -5.657      0.000      -0.002      -0.001
np.log(p)      0.0265      0.004      6.750      0.000       0.019       0.034
==============================================================================
Omnibus:                     1894.980   Durbin-Watson:                   1.975
```

```
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           237768.376
Skew:                           6.684   Prob(JB):                         0.00
Kurtosis:                      62.920   Cond. No.                         714.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[36]:
```
formulas = ["ami100 ~ vola + " for _ in range(6)]
formulas[0] += "mktcap"
formulas[1] += "vp"
formulas[2] += "np.log(mktcap)"
formulas[3] += "l_vola"
formulas[4] += "turnover"
formulas[5] += "turnover + np.log(p)"
```

[37]:
```
sm.ols(formulas[0], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
 ↪summary()
```

[37]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.028
Model:                            OLS   Adj. R-squared:                  0.027
Method:                 Least Squares   F-statistic:                     21.79
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           4.70e-10
Time:                        14:18:12   Log-Likelihood:                -2473.7
No. Observations:                1512   AIC:                             4953.
Df Residuals:                    1509   BIC:                             4969.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0477      0.062      0.765      0.445      -0.075       0.170
vola           6.7531      1.066      6.332      0.000       4.661       8.845
mktcap     -5.158e-06   4.16e-06     -1.241      0.215   -1.33e-05       3e-06
==============================================================================
Omnibus:                     2048.488   Durbin-Watson:                   2.037
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           420909.886
Skew:                           7.529   Prob(JB):                         0.00
Kurtosis:                      83.339   Cond. No.                     2.62e+05
==============================================================================
```

```
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.62e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[39]:
```
sm.ols(formulas[1], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
↪summary()
```

[39]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.029
Model:                            OLS   Adj. R-squared:                  0.027
Method:                 Least Squares   F-statistic:                     22.34
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           2.75e-10
Time:                        14:18:19   Log-Likelihood:                -2473.2
No. Observations:                1512   AIC:                             4952.
Df Residuals:                    1509   BIC:                             4968.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0513      0.062      0.822      0.411      -0.071       0.174
vola           6.7404      1.065      6.331      0.000       4.652       8.829
vp         -5.455e-07   3.38e-07     -1.615      0.106   -1.21e-06    1.17e-07
==============================================================================
Omnibus:                     2048.863   Durbin-Watson:                   2.038
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           421467.063
Skew:                           7.531   Prob(JB):                         0.00
Kurtosis:                      83.393   Cond. No.                     3.22e+06
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 3.22e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[40]:
```
sm.ols(formulas[2], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
↪summary()
```

[40]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.105
Model:                            OLS   Adj. R-squared:                  0.104
Method:                 Least Squares   F-statistic:                     88.71
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           3.78e-37
Time:                        14:18:23   Log-Likelihood:                -2411.2
No. Observations:                1512   AIC:                             4828.
Df Residuals:                    1509   BIC:                             4844.
Df Model:                           2
Covariance Type:            nonrobust
================================================================================
==
                    coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
Intercept         1.4203      0.134     10.569      0.000       1.157
1.684
vola              2.3582      1.092      2.160      0.031       0.216
4.500
np.log(mktcap)   -0.2237      0.019    -11.478      0.000      -0.262
-0.185
==============================================================================
Omnibus:                     2058.855   Durbin-Watson:                   2.048
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           456993.827
Skew:                           7.565   Prob(JB):                         0.00
Kurtosis:                      86.815   Cond. No.                         198.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[41]:

```
sm.ols(formulas[3], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
 ↪summary()
```

[41]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.027
Model:                            OLS   Adj. R-squared:                  0.026
Method:                 Least Squares   F-statistic:                     21.28
```

```
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           7.66e-10
Time:                        14:18:30   Log-Likelihood:                -2474.2
No. Observations:                1512   AIC:                             4954.
Df Residuals:                    1509   BIC:                             4970.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -0.3049      0.458     -0.666      0.506      -1.203       0.594
vola           8.3003      2.173      3.820      0.000       4.038      12.562
l_vola        -0.0853      0.114     -0.748      0.454      -0.309       0.138
==============================================================================
Omnibus:                     2048.463   Durbin-Watson:                   2.041
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           420060.081
Skew:                           7.530   Prob(JB):                         0.00
Kurtosis:                      83.255   Cond. No.                         233.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[42]:

```
sm.ols(formulas[4], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
 ↪summary()
```

[42]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.043
Model:                            OLS   Adj. R-squared:                  0.042
Method:                 Least Squares   F-statistic:                     33.98
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           3.69e-15
Time:                        14:18:37   Log-Likelihood:                -2461.9
No. Observations:                1512   AIC:                             4930.
Df Residuals:                    1509   BIC:                             4946.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1016      0.063      1.626      0.104      -0.021       0.224
vola           7.0429      1.054      6.684      0.000       4.976       9.110
turnover      -0.0091      0.002     -5.026      0.000      -0.013      -0.006
==============================================================================
```

```
Omnibus:                     2056.778   Durbin-Watson:                   2.051
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           430276.072
Skew:                           7.581   Prob(JB):                         0.00
Kurtosis:                      84.240   Cond. No.                         643.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[43]:
```
sm.ols(formulas[5], data.replace([np.inf, -np.inf], np.nan).dropna()).fit().
→summary()
```

[43]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.044
Model:                            OLS   Adj. R-squared:                  0.042
Method:                 Least Squares   F-statistic:                     23.13
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           1.22e-14
Time:                        14:18:42   Log-Likelihood:                -2461.2
No. Observations:                1512   AIC:                             4930.
Df Residuals:                    1508   BIC:                             4952.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1899      0.097      1.962      0.050    3.97e-05       0.380
vola           6.4369      1.169      5.505      0.000       4.143       8.731
turnover      -0.0087      0.002     -4.698      0.000      -0.012      -0.005
np.log(p)     -0.0345      0.029     -1.194      0.232      -0.091       0.022
==============================================================================
Omnibus:                     2056.461   Durbin-Watson:                   2.050
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           429910.948
Skew:                           7.579   Prob(JB):                         0.00
Kurtosis:                      84.205   Cond. No.                         716.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[54]:
```
data["fin"] = pd.get_dummies(data["gics"])[40]
sm.ols(formulas[3] + " + fin", data.replace([np.inf, -np.inf], np.nan).dropna()).
↪fit().summary()
```

[54]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.066
Model:                            OLS   Adj. R-squared:                  0.064
Method:                 Least Squares   F-statistic:                     35.68
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           2.82e-22
Time:                        14:25:30   Log-Likelihood:                -2443.4
No. Observations:                1512   AIC:                             4895.
Df Residuals:                    1508   BIC:                             4916.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -0.1014      0.450     -0.225      0.822      -0.984       0.781
vola           6.4719      2.142      3.021      0.003       2.270      10.674
l_vola        -0.0104      0.112     -0.093      0.926      -0.230       0.210
fin            0.6290      0.079      7.920      0.000       0.473       0.785
==============================================================================
Omnibus:                     2072.817   Durbin-Watson:                   2.063
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           473049.769
Skew:                           7.654   Prob(JB):                         0.00
Kurtosis:                      88.290   Cond. No.                         234.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[55]:
```
sm.ols(formulas[4] + " + fin", data.replace([np.inf, -np.inf], np.nan).dropna()).
↪fit().summary()
```

[55]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.078
Model:                            OLS   Adj. R-squared:                  0.076
Method:                 Least Squares   F-statistic:                     42.49
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           2.35e-26
```

```
Time:                        14:25:38   Log-Likelihood:                -2433.9
No. Observations:                1512   AIC:                             4876.
Df Residuals:                    1508   BIC:                             4897.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0023      0.063      0.037      0.971      -0.121       0.125
vola           6.4670      1.037      6.234      0.000       4.432       8.502
turnover      -0.0078      0.002     -4.367      0.000      -0.011      -0.004
fin            0.5964      0.079      7.549      0.000       0.441       0.751
==============================================================================
Omnibus:                     2079.300   Durbin-Watson:                   2.074
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           480755.833
Skew:                           7.696   Prob(JB):                         0.00
Kurtosis:                      88.989   Cond. No.                         645.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[58]:
```
reg = sm.ols(formulas[5] + " + fin", data.replace([np.inf, -np.inf], np.nan).
 ↪dropna()).fit()
reg.summary()
```

[58]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 ami100   R-squared:                       0.082
Model:                            OLS   Adj. R-squared:                  0.079
Method:                 Least Squares   F-statistic:                     33.58
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           7.26e-27
Time:                        14:27:52   Log-Likelihood:                -2430.7
No. Observations:                1512   AIC:                             4871.
Df Residuals:                    1507   BIC:                             4898.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1823      0.095      1.922      0.055      -0.004       0.368
vola           5.1590      1.158      4.456      0.000       2.888       7.430
turnover      -0.0068      0.002     -3.740      0.000      -0.010      -0.003
```

```
np.log(p)      -0.0727      0.029     -2.527      0.012      -0.129      -0.016
fin             0.6304      0.080      7.880      0.000       0.473       0.787
==============================================================================
Omnibus:                     2079.987   Durbin-Watson:                   2.074
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           483087.654
Skew:                           7.699   Prob(JB):                         0.00
Kurtosis:                      89.203   Cond. No.                         723.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

The impact of *fin* appears positive, indicating financial stocks were less liquid.

2.11.3 Part E

[81]:
```python
inv = lambda c : 1-c
data["nfin"] = data["fin"].apply(inv)

reg = sm.ols("spread ~ fin + nfin + vola:fin + vola:nfin + turnover:fin +␣
↪turnover:nfin + np.log(p):fin + np.log(p):nfin - 1",
             data).fit()
reg.summary()
```

[81]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                 spread   R-squared:                       0.169
Model:                            OLS   Adj. R-squared:                  0.166
Method:                 Least Squares   F-statistic:                     43.90
Date:                Mon, 13 Apr 2020   Prob (F-statistic):           1.05e-56
Time:                        14:40:59   Log-Likelihood:                 666.85
No. Observations:                1514   AIC:                            -1318.
Df Residuals:                    1506   BIC:                            -1275.
Df Model:                           7
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
fin                0.0033      0.035      0.094      0.925      -0.065
```

```
0.072
nfin               0.0269      0.013      2.011      0.044       0.001
0.053
vola:fin           1.4296      0.305      4.684      0.000       0.831
2.028
vola:nfin          0.2226      0.174      1.280      0.201      -0.118
0.564
turnover:fin      -0.0099      0.001     -8.596      0.000      -0.012
-0.008
turnover:nfin     -0.0006      0.000     -2.467      0.014      -0.001
-0.000
np.log(p):fin      0.0762      0.011      7.140      0.000       0.055
0.097
np.log(p):nfin     0.0109      0.004      2.742      0.006       0.003
0.019
==============================================================================
Omnibus:                     1851.623   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           231906.399
Skew:                           6.395   Prob(JB):                         0.00
Kurtosis:                      62.267   Cond. No.                     1.45e+03
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.45e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

[82]:
```
hypothesis = "(vola:fin = vola:nfin)"
reg.f_test(hypothesis)
```

[82]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[11.80821715]]), p=0.000605737283498603, df_denom=1.51e+03,
df_num=1>
```

[83]:
```
hypothesis = "(turnover:fin = turnover:nfin)"
reg.f_test(hypothesis)
```

[83]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[62.50863367]]), p=5.088683210731206e-15, df_denom=1.51e+03,
df_num=1>
```

[84]:
```
hypothesis = "(np.log(p):fin = np.log(p):nfin)"
reg.f_test(hypothesis)
```

[84]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[32.96022594]]), p=1.1356962949105402e-08, df_denom=1.51e+03,
df_num=1>
```

[85]:
```python
hypothesis = "(vola:fin = vola:nfin), (turnover:fin = turnover:nfin), (np.log(p):
 ↪fin = np.log(p):nfin)"
reg.f_test(hypothesis)
```

[85]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[28.93117199]]), p=3.469463998723971e-18, df_denom=1.51e+03,
df_num=3>
```

3 Chapter 4

3.1 Question 1

3.1.1 Part A

The parameter λ is derived:

$$\begin{aligned} p(q) &= \mu + \frac{cov(v,q)}{var(q)} q && (55) \\ &= \mu + \frac{\beta\sigma_v^2}{\beta(\sigma_v^2 + \sigma_\eta^2) + \sigma_u^2} q && (56) \end{aligned}$$

An increase in the informed investors error increases σ_η^2, increasing market depth. Shifting the response curve up and to the left.

3.1.2 Part B

$$\begin{aligned} E[v|s] &= \mu + \frac{cov(v,s)}{var(s)}(s - \mu) && (57) \\ &= \mu + \frac{\sigma_v^2}{\sigma_v^2 + \sigma_\nu^2} && (58) \end{aligned}$$

The investor wishes to maximise his value:

$$\max_x E[x * (v - p)|s] = \max_x xE[v - p|s] \tag{59}$$
$$= \max_x x(E[v|s] - \mu - \lambda x) \tag{60}$$
$$= \max_x x(\mu + \frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2}(s - \mu) - \mu - \lambda x) \tag{61}$$
$$= \max_x x(\frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2}(s - \mu) - \lambda x) \tag{62}$$
$$0 \equiv \frac{d}{dx} x(\frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2}(s - \mu) - \lambda x) \tag{63}$$
$$= \frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2}(s - \mu) - 2\lambda x \tag{64}$$
$$x = \frac{1}{2\lambda} \frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2}(s - \mu) \tag{65}$$
$$\beta = \frac{1}{2\lambda} \frac{\sigma_v^2}{\sigma_v^2 + \sigma_\eta^2} \tag{66}$$

Thus beta is decreasing in σ_η^2

3.1.3 Part C

$$\lambda = \frac{\beta\sigma_v^2}{\beta(\sigma_v^2+\sigma_\eta^2)+\sigma_u^2} \tag{67}$$

$$\beta = \frac{1}{2\lambda}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2} \tag{68}$$

$$= \frac{1}{2\frac{\beta\sigma_v^2}{\beta(\sigma_v^2+\sigma_\eta^2)+\sigma_u^2}}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2} \tag{69}$$

$$= \frac{\beta(\sigma_v^2+\sigma_\eta^2)+\sigma_u^2}{2\beta\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2} \tag{70}$$

$$\beta^2 = \frac{\beta(\sigma_v^2+\sigma_\eta^2)+\sigma_u^2}{2\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2} \tag{71}$$

$$= \frac{\beta\sigma_v^2+\beta\sigma_\eta^2+\sigma_u^2}{2\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2} \tag{72}$$

$$= \left(\frac{\beta\sigma_v^2+\beta\sigma_\eta^2}{2\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}\right)+\left(\frac{\sigma_u^2}{2\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}\right) \tag{73}$$

$$= \left(\frac{\beta}{2}\right)+\left(\frac{\sigma_u^2}{2\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}\right) \tag{74}$$

$$= \left(\frac{\sigma_u^2}{\sigma_v^2}\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}\right) \tag{75}$$

$$\beta = \frac{\sigma_u}{\sigma_v}\sqrt{\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}} \tag{76}$$

$$\lambda = \frac{\sigma_v}{2\sigma_u}\sqrt{\frac{\sigma_v^2}{\sigma_v^2+\sigma_\eta^2}} \tag{77}$$

3.2 Question 2

3.2.1 Part A

Set $u = u_1 + u_2$, $\sigma_u = \sqrt{\sigma_{u1}^2+\sigma_{u2}^2}$

$$E[v|q] = \mu + \frac{\beta\sigma_v^2}{\beta\sigma_v^2 + \sigma_{u1}^2 + \sigma_{u2}^2} \tag{78}$$

$$x = \frac{1}{2\lambda}(v - \mu) \tag{79}$$

$$\lambda = \frac{\sigma_v}{2\sqrt{\sigma_{u1}^2 + \sigma_{u2}^2}} \tag{80}$$

$$\beta = \frac{\sqrt{\sigma_{u1}^2 + \sigma_{u2}^2}}{\sigma_v} \tag{81}$$

3.2.2 Part B

$$E[v|q] = \mu + \frac{\beta\sigma_v^2}{\beta\sigma_v^2 + \sigma_{u2}^2} \tag{82}$$

$$x = \frac{1}{2\lambda}(v - \mu) \tag{83}$$

$$\lambda = \frac{\sigma_v}{2\sigma_{u2}} \tag{84}$$

$$\beta = \frac{\sigma_{u2}}{\sigma_v} \tag{85}$$

3.2.3 Part C

The difference depends on the ratio of $\sigma_{u1} : \sigma_{u2}$. If the latter is larger then depth and aggresiveness is increased, else, the opposite.

3.3 Question 3

3.3.1 Part A

From each traders perspective:

$$
\begin{aligned}
p &= \mu + \lambda(x_i + (N-1)\beta(v-\mu) + u) && (86)\\
E[p] &= \mu + \lambda(x_i + (N-1)\beta(v-\mu)) && (87)\\
E[x_i(v-p)] &= E[x_i(v-\mu-\lambda(x_i+(N-1)\beta(v-\mu)))] && (88)\\
&= E[x_i((v-\mu)(1-(N-1)\beta) - \lambda x_i)] && (89)\\
\frac{d}{dx_i} x_i((v-\mu)(1-(N-1)\beta) - \lambda x_i) &= (v-\mu)(1-(N-1)\beta) - 2\lambda x_i && (90)\\
0 &= (v-\mu)(1-(N-1)\beta) - 2\lambda x_i && (91)\\
2\lambda x_i &= (v-\mu)(1-(N-1)\beta) && (92)\\
x_i &= \frac{(1-(N-1)\beta)}{2\lambda}(v-\mu) && (93)\\
\beta &= \frac{(1-(N-1)\beta)}{2\lambda} && (94)\\
2\lambda\beta &= 1-(N-1)\beta && (95)\\
(2\lambda + (N-1))\beta &= 1 && (96)\\
\beta &= \frac{1}{2\lambda + (N-1)} && (97)
\end{aligned}
$$

Thus β is decreasing in N

3.3.2 Part B

$$p(q) = \mu + \lambda q \tag{98}$$
$$= \mu + \frac{\beta\sigma_v^2}{\beta\sigma_v^2 + \sigma_u^2} \tag{99}$$
$$\frac{1}{\lambda} = \beta + \frac{\sigma_u^2}{\beta\sigma_v^2} \tag{100}$$
$$= \frac{1}{2\lambda + (N-1)} + \frac{\sigma_u^2}{\frac{1}{2\lambda+(N-1)}\sigma_v^2} \tag{101}$$
$$= \frac{1}{2\lambda + (N-1)} + \frac{(2\lambda + (N-1))\sigma_u^2}{\sigma_v^2} \tag{102}$$
$$\frac{2\lambda + (N-1)}{\lambda} - 1 = \frac{(2\lambda + (N-1))^2\sigma_u^2}{\sigma_v^2} \tag{103}$$
$$\frac{\lambda + (N-1)}{\lambda} = \frac{(2\lambda + (N-1))^2\sigma_u^2}{\sigma_v^2} \tag{104}$$
$$1 + \frac{(N-1)}{\lambda} = \frac{(2\lambda + (N-1))^2\sigma_u^2}{\sigma_v^2} \tag{105}$$
$$\lambda + (N-1) = \frac{\lambda(2\lambda\sigma_u + (N-1)\sigma_u)^2}{\sigma_v^2} \tag{106}$$
$$\lambda\sigma_v^2 + (N-1)\sigma_v^2 = \lambda(2\lambda\sigma_u + (N-1)\sigma_u)^2 \tag{107}$$
$$\lambda\sigma_v^2 + (N-1)\sigma_v^2 = \lambda(4\lambda^2\sigma_u^2 + 4\lambda\sigma_u^2(N-1) + (N-1)^2\sigma_u^2) \tag{108}$$
$$(N-1+\lambda)\sigma_v^2 = 4\lambda^2\sigma_u^2(N-1+\lambda) + \lambda(N-1)^2\sigma_u^2 \tag{109}$$
$$(N-1+\lambda)(\sigma_v^2 - 4\lambda^2\sigma_u^2) = \lambda(N-1)^2\sigma_u^2 \tag{110}$$
$$\frac{1}{\lambda} = \frac{N+1}{\sqrt{N}}\frac{\sigma_u}{\sigma_v} \tag{111}$$

Thus market depth is increasing with N.

3.3.3 Part C

As above.

3.3.4 Part D

$$E[x_i(v-p)] = E[x_i((v-\mu)(1-(N-1)\beta) - \lambda x_i)] \tag{112}$$
$$= \frac{1}{(N+1)\lambda} \tag{113}$$

3.4 Question 4

3.4.1 Part A

$$\begin{aligned}
var(v-p) &= var(v-\mu-\lambda(x+u)) && (114)\\
&= var(v-\mu-\lambda(\beta(v-\mu)+u) && (115)\\
&= var(v-\mu-\lambda\beta(v-\mu)-\lambda u) && (116)\\
&= var((1-\lambda\beta)(v-\mu)-\lambda u) && (117)\\
&= (1-\lambda\beta)^2\sigma_v^2+\lambda^2\sigma_u^2 && (118)\\
&= (1-\frac{\beta^2\sigma_v^2}{\beta^2\sigma_v^2+\sigma_u^2})^2\sigma_v^2+(\frac{\beta\sigma_v^2}{\beta^2\sigma_v^2+\sigma_u^2})^2\sigma_u^2 && (119)\\
&= (\frac{\sigma_u^2}{\beta^2\sigma_v^2+\sigma_u^2})^2\sigma_v^2+(\frac{\beta\sigma_v^2}{\beta^2\sigma_v^2+\sigma_u^2})^2\sigma_u^2 && (120)\\
&= (\frac{\sigma_u^2\sigma_v}{\beta^2\sigma_v^2+\sigma_u^2})^2+(\frac{\beta\sigma_v^2\sigma_u}{\beta^2\sigma_v^2+\sigma_u^2})^2 && (121)\\
&= (\beta^2\sigma_v^2+\sigma_u^2)(\frac{\sigma_u\sigma_v}{\beta^2\sigma_v^2+\sigma_u^2})^2 && (122)\\
&= \frac{\sigma_u^2\sigma_v^2}{\beta^2\sigma_v^2+\sigma_u^2} && (123)
\end{aligned}$$

Trader's aggresivenes serves to correct the difference $v-p$, minimising its variance.

3.4.2 Part B

$$\begin{aligned}
var(p-u) &= var(\mu+\lambda(x+u)-u) && (124)\\
&= var(\mu+\lambda x-(1-\lambda)u) && (125)\\
&= var(\mu+\lambda\beta(v-\mu)-(1-\lambda)u) && (126)\\
&= var((1-\lambda\beta)\mu+\lambda\beta v-(1-\lambda)u) && (127)\\
&= (\frac{\beta\sigma_v^2}{\beta^2\sigma_v^2+\sigma_u^2})^2(\beta^2\sigma_v^+\sigma_u^2) && (128)\\
&= \frac{\beta^2\sigma_v^4}{\beta^2\sigma_v^2+\sigma_u^2} && (129)
\end{aligned}$$

4 Chapter 5

[1]:
```
import pandas as pd
data = pd.read_excel("data_2/Ch5_AGF_data.xls", header=1, index_col="t")
data.head()
```

[1]:
```
       Time  Tradesize  Traprice  Tradedir    Bid    Ask
t
1  09:06:04         20   66.7000        -1  66.90  67.00
2  09:06:11         25   66.6360        -1  66.65  66.70
3  09:06:26         18   66.6000        -1  66.60  66.65
4  09:07:18        273   66.4163        -1  66.50  66.55
5  09:07:36         27   66.5500         1  66.15  66.55
```

[2]:
```python
data['l_Traprice'] = data['Traprice'].shift()
data['l_Tradedir'] = data['Tradedir'].shift()
data['d_Traprice'] = data['Traprice'] - data['l_Traprice']

import statsmodels.formula.api as sm

reg = sm.ols("d_Traprice ~ Tradedir + l_Tradedir - 1", data).fit()
reg.summary()
```

```
/usr/local/lib/python3.7/site-packages/statsmodels/tools/_testing.py:19:
FutureWarning: pandas.util.testing is deprecated. Use the functions in the
public API at pandas.testing instead.
  import pandas.util.testing as tm
/usr/local/lib/python3.7/site-packages/statsmodels/compat/pandas.py:23:
FutureWarning: The Panel class is removed from pandas. Accessing it from the
top-level namespace will also be removed in the next version
  data_klasses = (pandas.Series, pandas.DataFrame, pandas.Panel)
```

[2]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:             d_Traprice   R-squared (uncentered):
0.333
Model:                            OLS   Adj. R-squared (uncentered):
0.330
Method:                 Least Squares   F-statistic:
128.7
Date:                Wed, 13 May 2020   Prob (F-statistic):
4.61e-46
Time:                        10:06:35   Log-Likelihood:
587.57
No. Observations:                 518   AIC:
-1171.
Df Residuals:                     516   BIC:
-1163.
Df Model:                           2
Covariance Type:            nonrobust
```

```
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Tradedir       0.0587      0.004     15.261      0.000       0.051       0.066
l_Tradedir    -0.0437      0.004    -11.355      0.000      -0.051      -0.036
==============================================================================
Omnibus:                      122.063   Durbin-Watson:                   2.516
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3596.694
Skew:                          -0.225   Prob(JB):                         0.00
Kurtosis:                      15.901   Cond. No.                         1.64
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[3]:
```python
data['mid'] = (data['Bid'] + data['Ask']) / 2
data['d_mid'] = data['mid'] - data['mid'].shift()
data['ld_mid'] = data['d_mid'].shift()
data['l2d_mid'] = data['d_mid'].shift(2)
data['l3d_mid'] = data['d_mid'].shift(3)
data['l2_Tradedir'] = data['Tradedir'].shift(2)
data['l3_Tradedir'] = data['Tradedir'].shift(3)

formulas = ["d_mid ~ ld_mid + l2d_mid + l3d_mid + l_Tradedir + l2_Tradedir +␣
 ↪l3_Tradedir - 1",
            "Tradedir ~ ld_mid + l2d_mid + l3d_mid + l_Tradedir + l2_Tradedir +␣
 ↪l3_Tradedir - 1"]
reg = [sm.ols(formulas[f], data).fit() for f in range(2)]
```

[4]:
```python
reg[0].summary()
```

[4]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:                  d_mid   R-squared (uncentered):
0.129
Model:                            OLS   Adj. R-squared (uncentered):
0.119
Method:                 Least Squares   F-statistic:
12.61
Date:                Wed, 13 May 2020   Prob (F-statistic):
2.78e-13
Time:                        10:06:36   Log-Likelihood:
```

```
759.40
No. Observations:                 515   AIC:
-1507.
Df Residuals:                     509   BIC:
-1481.
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
ld_mid        -0.1481      0.044     -3.358      0.001      -0.235      -0.061
l2d_mid       -0.0702      0.044     -1.592      0.112      -0.157       0.016
l3d_mid        0.0805      0.041      1.954      0.051      -0.000       0.162
l_Tradedir     0.0142      0.003      5.057      0.000       0.009       0.020
l2_Tradedir    0.0075      0.003      2.437      0.015       0.001       0.013
l3_Tradedir    0.0027      0.003      0.917      0.360      -0.003       0.008
==============================================================================
Omnibus:                      127.272   Durbin-Watson:                   2.013
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1950.879
Skew:                           0.612   Prob(JB):                         0.00
Kurtosis:                      12.456   Cond. No.                         26.2
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

[5]:
```
reg[1].summary()
```

[5]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:               Tradedir   R-squared (uncentered):
0.261
Model:                            OLS   Adj. R-squared (uncentered):
0.252
Method:                 Least Squares   F-statistic:
29.97
Date:                Wed, 13 May 2020   Prob (F-statistic):
8.24e-31
Time:                        10:06:37   Log-Likelihood:
-652.84
No. Observations:                 515   AIC:
1318.
```

```
Df Residuals:                     509   BIC:
1343.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
ld_mid         0.5068      0.684      0.741      0.459      -0.838       1.851
l2d_mid        0.7703      0.684      1.126      0.261      -0.574       2.115
l3d_mid       -0.1092      0.640     -0.171      0.865      -1.366       1.148
l_Tradedir     0.3588      0.044      8.227      0.000       0.273       0.444
l2_Tradedir    0.0502      0.048      1.056      0.291      -0.043       0.144
l3_Tradedir    0.2029      0.046      4.443      0.000       0.113       0.293
==============================================================================
Omnibus:                       31.966   Durbin-Watson:                   1.998
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               12.909
Skew:                           0.109   Prob(JB):                      0.00157
Kurtosis:                       2.256   Cond. No.                         26.2
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```

4.1 Question 3

4.1.1 Part A

[6]:
```
data = pd.read_excel("data_2/Ch5_ex3_data.xls", index_col='t')
fig = data['inventories'].plot() # Yes there is evidence of mean revesion
```

[7]:
```
data.head()
```

[7]:
```
    price  tradesize  signedtr  inventories  Delta_p  Delta_q  Delta_d  \
t
1  1.4794      -1.00        -1         1.00      NaN      NaN      NaN
2  1.4797      -2.00        -1         3.00   0.0003    -1.00      0.0
3  1.4795     -28.00        -1         1.00  -0.0002   -26.00      0.0
4  1.4794      -0.50        -1        -1.50  -0.0001    27.50      0.0
5  1.4790      -0.75        -1        -0.75  -0.0004    -0.25      0.0

   Delta_i      q
t
1      NaN    NaN
2     2.00  -2.00
```

```
3    -2.00 -28.00
4    -2.50  -0.50
5     0.75  -0.75
```

4.1.2 Part B

[8]:
```
data['Delta_z'] = -data['q'].shift()
formula = "Delta_p ~ q + Delta_q + Delta_z + Delta_d - 1"
reg = sm.ols(formula, data).fit()
reg.summary()
```

[8]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
================================================================================
=======
Dep. Variable:                Delta_p   R-squared (uncentered):
0.214
Model:                            OLS   Adj. R-squared (uncentered):
0.212
Method:                 Least Squares   F-statistic:
75.54
Date:                Wed, 13 May 2020   Prob (F-statistic):
3.30e-43
Time:                        10:06:41   Log-Likelihood:
5298.3
No. Observations:                 833   AIC:
-1.059e+04
Df Residuals:                     830   BIC:
-1.058e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
q           1.582e-05   2.14e-06      7.400      0.000    1.16e-05       2e-05
Delta_q     3.901e-06   1.97e-06      1.983      0.048    3.98e-08    7.76e-06
Delta_z    -1.192e-05   2.13e-06     -5.595      0.000   -1.61e-05   -7.74e-06
Delta_d        0.0001   1.55e-05      7.564      0.000    8.69e-05       0.000
==============================================================================
Omnibus:                      227.993   Durbin-Watson:                   2.163
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7896.812
Skew:                           0.517   Prob(JB):                         0.00
Kurtosis:                      18.048   Cond. No.                     1.63e+16
==============================================================================
```

```
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The smallest eigenvalue is 2.38e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
"""
```

4.2 Question 5

[9]:
```python
data = pd.read_excel("data_2/Ch5_AGF_data.xls", header=1, index_col="t")
data.head()
```

[9]:
```
       Time  Tradesize  Traprice  Tradedir    Bid    Ask
t
1  09:06:04         20   66.7000        -1  66.90  67.00
2  09:06:11         25   66.6360        -1  66.65  66.70
3  09:06:26         18   66.6000        -1  66.60  66.65
4  09:07:18        273   66.4163        -1  66.50  66.55
5  09:07:36         27   66.5500         1  66.15  66.55
```

[13]:
```python
data['delta_p'] = data['Traprice'] - data['Traprice'].shift()
data['delta_q'] = data['Tradesize'] - data['Tradesize'].shift()
data['delta_d'] = data['Tradedir'] - data['Tradedir'].shift()
data.head()
```

[13]:
```
       Time  Tradesize  Traprice  Tradedir    Bid    Ask  delta_p  delta_q  \
t
1  09:06:04         20   66.7000        -1  66.90  67.00      NaN      NaN
2  09:06:11         25   66.6360        -1  66.65  66.70  -0.0640      5.0
3  09:06:26         18   66.6000        -1  66.60  66.65  -0.0360     -7.0
4  09:07:18        273   66.4163        -1  66.50  66.55  -0.1837    255.0
5  09:07:36         27   66.5500         1  66.15  66.55   0.1337   -246.0

   detla_d  delta_d
t
1      NaN      NaN
2      0.0      0.0
3      0.0      0.0
4      0.0      0.0
5      2.0      2.0
```

[14]:
```python
import statsmodels.formula.api as sm
formula = "delta_p ~ Tradedir + Tradesize + delta_d + delta_q"
reg = sm.ols(formula, data).fit()
reg.summary()
```

[14]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                delta_p   R-squared:                       0.337
Model:                            OLS   Adj. R-squared:                  0.332
Method:                 Least Squares   F-statistic:                     65.28
Date:                Wed, 13 May 2020   Prob (F-statistic):           1.28e-44
Time:                        10:18:30   Log-Likelihood:                 589.34
No. Observations:                 518   AIC:                            -1169.
Df Residuals:                     513   BIC:                            -1147.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -0.0048      0.005     -1.039      0.300      -0.014       0.004
Tradedir       0.0155      0.004      3.848      0.000       0.008       0.023
Tradesize   9.583e-06   5.36e-06      1.789      0.074   -9.39e-07    2.01e-05
delta_d        0.0432      0.004     11.205      0.000       0.036       0.051
delta_q    -6.492e-06   4.25e-06     -1.528      0.127   -1.48e-05    1.86e-06
==============================================================================
Omnibus:                      121.720   Durbin-Watson:                   2.534
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3725.170
Skew:                          -0.185   Prob(JB):                         0.00
Kurtosis:                      16.132   Cond. No.                     1.79e+03
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.79e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

4.2.1 Part A

For a very small transaction the spread is equal to the quoted spread.

4.2.2 Part B

[16]:
```
hypothesis_0 = "(Tradedir = 0), (delta_q = 0)"
reg.f_test(hypothesis_0)
```

[16]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[8.42659165]]), p=0.000250729896174072, df_denom=513,
df_num=2>
```

We can reject the null hypothesis that both d_t and Δq_t are 0

4.2.3 Part C

[18]:
```
hypothesis_1 = "(Tradedir = 0), (delta_d = 0)"
reg.f_test(hypothesis_1)
```

[18]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[127.67465247]]), p=9.983823568521155e-46, df_denom=513,
df_num=2>
```

[19]:
```
hypothesis_2 = "(Tradesize = 0), (delta_q = 0)"
reg.f_test(hypothesis_2)
```

[19]:
```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[1.7337009]]), p=0.17766277190371602, df_denom=513, df_num=2>
```

The null hypothesis that trade size effects are not significant cannot be rejected.

4.2.4 Part D

In chapter 2, we explored the concept of price movement being a function of trade volumen, however the evidence here does not support trade size effects being statistically significant effecters of price change.

4.3 Chapter 8

4.3.1 Question 1

In this new situation the four order patterns are:

$$p(b,s) = \frac{1-\pi}{4} \tag{130}$$
$$p(s,b) = \frac{1-\pi}{4} \tag{131}$$
$$p(b,b) = \frac{\pi}{2} + \frac{1-\pi}{4} \tag{132}$$
$$= \frac{1+\pi}{4} \tag{133}$$
$$p(s,s) = \frac{1+\pi}{4} \tag{134}$$
$$E[v|b,s] = \mu \tag{135}$$
$$E[v|s,b] = \mu \tag{136}$$
$$E[v|b,b] = \frac{\frac{\pi}{2}}{\frac{1+\pi}{4}} v_H + \frac{\frac{1-\pi}{4}}{\frac{1+\pi}{4}} \mu \tag{137}$$
$$= \frac{2\pi}{1+\pi} v_H + \frac{1-\pi}{1+\pi} \mu \tag{138}$$
$$E[v|s,s] = \frac{2\pi}{1+\pi} v_L + \frac{1-\pi}{1+\pi} \mu \tag{139}$$
$$\tag{140}$$

This does not effect the bid and ask prices of dealers in opaque markets, which are still:

$$a^O = \mu + \pi(v_H - \mu)$$
$$b^O = \mu - \pi(\mu - v_L)$$

However in in the case of two buy orders the ask will be set to that shown in equation (9) and in the case of two sell orders the bid will be set to that in equation (10).

4.3.2 Question 2

$$E[(p_1^T - v)^2] = (1-\pi^2)(\frac{v_H - v_L}{2})^2$$

The second period transparant devation depends on whether or not an informed trader is present in the market.

$$E[(p_2^T - v)^2] = \pi 0 + (1-\pi)(\frac{v_H - v_L}{2})^2$$
$$\frac{E[(p_1^T - v)^2]}{2} + \frac{E[(p_2^T - v)^2]}{2} = \frac{(1-\pi^2)(\frac{v_H - v_L}{2})^2}{2} + \frac{(1-\pi)(\frac{v_H - v_L}{2})^2}{2}$$
$$= \frac{(\frac{v_H - v_L}{2})^2}{2}((1-\pi^2) + (1-\pi))$$

For the opaque regime, Eqn. (8.10) gives:

$$\begin{aligned}
s_1^O &= (2\pi - 1)(v_H - v_L) \\
E[(p_1^O - v)^2)] &= \left(\frac{s_1^O}{2}\right)^2 \\
&= (2\pi - 1)^2 (\frac{v_H - v_L}{2})^2
\end{aligned}$$

Using the fact that expected profits are 0 ($TC^O = \pi(v_H - v_L)$), we can calulate the spread in the second period.

$$\begin{aligned}
\frac{1}{2}s_1^O + \frac{1}{2}s_2^O &= TC^O \\
s_1^O + s_2^O &= 2TC^O \\
(2\pi - 1)(v_H - v_L) + s_2^O &= 2\pi(v_H - v_L) \\
s_2^O &= 2\pi(v_H - v_L) - (2\pi - 1)(v_H - v_L) \\
&= (2\pi - 2\pi + 1)(v_H - v_L) \\
&= v_H - v_L \\
E[(p_2^O - v)^2] &= \left(\frac{s_2^O}{2}\right)^2 \\
&= \left(\frac{v_H - v_L}{2}\right)^2 \\
\frac{E[(p_1^O - v)^2]}{2} + \frac{E[(p_2^O - v)^2]}{2} &= \frac{(2\pi - 1)^2 (\frac{v_H - v_L}{2})^2}{2} + \frac{(\frac{v_H - v_L}{2})^2}{2} \\
&= \frac{(2\pi - 1)^2 + 1}{2} \left(\frac{v_H - v_L}{2}\right)^2
\end{aligned}$$

5 Chapter 9

5.1 Question 1

5.1.1 Part A

$$1 + R = \frac{\mu_{t+1}}{\mu_t} + d\mu_t$$

Part B

$$\begin{aligned}
1 + R &= \frac{\mu_{t+1}(1 + \frac{s}{2}) + d\mu_t}{\mu_t(1 - \frac{s}{2})} \\
&= (1 + r)\frac{1 + \frac{s}{2}}{1 - \frac{s}{2}} - \frac{d\frac{s}{2}}{1 - \frac{s}{2}}
\end{aligned}$$

Part C

$$
\begin{aligned}
R - r &= (1+R) - (1+r) \\
&= (1+r)\frac{1+\frac{s}{2}}{1-\frac{s}{2}} - \frac{d\frac{s}{2}}{1-\frac{s}{2}} - (1+r) \\
&= (1+r)\left(\frac{1+\frac{s}{2}}{1-\frac{s}{2}} - \frac{d\frac{s}{2}}{1-\frac{s}{2}} - 1\right) \\
&= (1+r)\left(\frac{1+\frac{s}{2}}{1-\frac{s}{2}} - \frac{d\frac{s}{2}}{1-\frac{s}{2}} - \frac{1-\frac{s}{2}}{1-\frac{s}{2}}\right) \\
&= (1+r)\left(\frac{s-d\frac{s}{2}}{1-\frac{s}{2}}\right) \\
&= s\frac{(1+r)(1-\frac{d}{2})}{1-\frac{s}{2}}
\end{aligned}
$$

5.2 Question 2

5.2.1 Part A

$$
\begin{aligned}
1 + r &= \frac{\mu_{t+1}(1-\frac{s}{2}) + \bar{D}}{\mu_t(1+\frac{s}{2})} \\
&= \frac{\mu_t(1-\frac{s}{2}) + \bar{D}}{\mu_t(1+\frac{s}{2})} \qquad \text{Because expected dividends and irr are constant} \\
\mu_t(1+\frac{s}{2})(1+r) &= \mu_t(1-\frac{s}{2}) + \bar{D} \\
\bar{D} &= \mu_t(1+\frac{s}{2})(1+r) - \mu_t(1-\frac{s}{2}) \\
&= \mu_t\left((1+\frac{s}{2})(1+r) - (1-\frac{s}{2})\right) \\
&= \mu_t\left(1-\frac{s}{2}\right)\left((1+r)\frac{1+\frac{s}{2}}{1-\frac{s}{2}} - 1\right) \\
\mu_t &= \frac{\bar{D}}{(1-\frac{s}{2})\left((1+r)\frac{1+\frac{s}{2}}{1-\frac{s}{2}} - 1\right)} \\
\mu_t &= \frac{\bar{D}}{(1-\frac{s}{2})R}
\end{aligned}
$$

Part B

$$
\begin{aligned}
E[r_t - r] &= E[r_t] - r \\
&= 0
\end{aligned}
$$

Part C The new expected price is

$$
\mu_t^* = \frac{\bar{D}}{(1-\frac{s^*}{2})\left((1+r)\frac{1+\frac{s^*}{2}}{1-\frac{s^*}{2}} - 1\right)}
$$

$$
\begin{aligned}
1 + r_{\tau-1} &= \frac{\mu(1 - \frac{s^*}{2}) + \bar{D}}{\mu_t(1 + \frac{s}{2})} \\
E[r_{\tau-1} - r] &= \frac{\mu(1 - \frac{s^*}{2}) + \bar{D}}{\mu_t(1 + \frac{s}{2})} - \frac{\mu(1 - \frac{s}{2}) + \bar{D}}{\mu_t(1 + \frac{s}{2})} \\
&= \frac{\mu(1 - \frac{s^*}{2}) + \bar{D} - \mu(1 - \frac{s}{2}) - \bar{D}}{\mu_t(1 + \frac{s}{2})} \\
&= \frac{\mu\frac{s}{2} - \mu\frac{s^*}{2}}{\mu_t(1 + \frac{s}{2})} \\
&= \frac{s - s^*}{2 + s}
\end{aligned}
$$

5.2.2 Part D

Returns of a security are not determined by it's liquidity so there should be no difference. ## Question 3

$$
E[R_j] = \frac{E[s_j]}{h} + \lambda_M \left(\beta_1 - \frac{\beta_3 + \beta_4 - \frac{\beta_2}{h}}{h} \right)
$$

5.3 Question 4

Both supply and demand by arbitrageurs are increased by a factor of K

$$
\begin{aligned}
y(P_{A1}) + K \int_0^{\hat{\varphi}} \frac{i}{K} di &= K(1 - \hat{\varphi}) \\
1 + \delta(P_{A1} - V) + K \int_0^{\hat{\varphi}} \frac{i}{K} di &= K(1 - \hat{\varphi}) \\
1 + \delta(P_{A1} - V) + \frac{\hat{\varphi}^2}{2} &= K(1 - \hat{\varphi}) \\
\delta V - \delta P_{A1} &= 1 + \frac{\hat{\varphi}^2}{2} - K(1 - \hat{\varphi}) \\
V - P_{A1} &= \frac{1}{\delta} + \frac{1}{\delta}\frac{\hat{\varphi}^2}{2} - \frac{K}{\delta}(1 - \hat{\varphi}) \\
\frac{\partial(V - P_{A1})}{\partial K} &= -\frac{1 - \hat{\varphi}}{\delta} \\
&< 0
\end{aligned}
$$

Thus as the mass of arbitrageurs gets larger, mispricing at time 1 is reduced.

5.4 Question 5

5.4.1 Part A

The primary requirements of the model are:

$$\begin{aligned}
\hat{\varphi} &= \frac{M_0 - \kappa M_1}{\kappa M_1} \\
y(P_{A1}) + \int_0^{\hat{\varphi}\varphi(i)di} &= (1-\varphi) \\
y(P_{A0}) &= \varphi
\end{aligned}$$

These identities give the result

$$M_1 = \frac{1}{\delta}(\hat{\varphi} - \frac{\hat{\varphi}^2}{2})$$

At time 0, market clearing requires $y(P_{A0}) = \hat{\varphi}$. Subsituting into th given equation:

$$\begin{aligned}
\hat{\varphi} &= 1 - \delta_0(V - P_{A0}) \\
&= 1 - \delta_0 M_0 \\
M_0 &= \frac{1-\hat{\varphi}}{\delta_0}
\end{aligned}$$

5.4.2 Part B

$$\begin{aligned}
M_0 &= \frac{1-\hat{\varphi}}{0.4} \\
M_1 &= \frac{\hat{\varphi} - \frac{\hat{\varphi}^2}{2}}{0.1} \\
\hat{\varphi} &= \frac{M_0 - 0.1M_1}{0.1M_1} \\
&= \frac{\frac{1-\hat{\varphi}}{0.4} - \hat{\varphi} + \frac{\hat{\varphi}^2}{2}}{\hat{\varphi} - \frac{\hat{\varphi}^2}{2}} \\
&= \frac{2.5 - 3.5\hat{\varphi} + \frac{\hat{\varphi}^2}{2}}{\hat{\varphi} - \frac{\hat{\varphi}^2}{2}} \\
&= \frac{5 - 7\hat{\varphi} + \hat{\varphi}^2}{2\hat{\varphi} - \hat{\varphi}^2} \\
2\hat{\varphi}^2 - \hat{\varphi}^3 &= 5 - 7\hat{\varphi} + \hat{\varphi}^2 \\
0 &= \hat{\varphi}^3 - \hat{\varphi}^2 - 7\hat{\varphi} + 5
\end{aligned}$$

[21]:
```
import numpy as np
coeff = [1, -1, -7, 5]
phi = np.roots(coeff)[-1]
M0 = (1 - phi) / 0.4
M1 = (phi - (phi ** 2) / 2) / 0.1
print(phi, M0, M1)
```

```
0.693225133872129 0.7669371653196774 4.529445907561134
```

Repeating for the case kappa = 0.3

$$
\begin{aligned}
\hat{\varphi} &= \frac{M_0 - 0.3M_1}{0.3M_1} \\
&= \frac{\frac{1-\hat{\varphi}}{0.4} - 3(\hat{\varphi} - \frac{\hat{\varphi}^2}{2})}{3(\hat{\varphi} - \frac{\hat{\varphi}^2}{2})} \\
&= \frac{2.5 - 5.5\hat{\varphi} + 3\frac{\hat{\varphi}^2}{2}}{3\hat{\varphi} - 3\frac{\hat{\varphi}^2}{2}} \\
&= \frac{5 - 11\hat{\varphi} + 3\hat{\varphi}^2}{6\hat{\varphi} - 3\hat{\varphi}^2} \\
6\hat{\varphi}^2 - 3\hat{\varphi}^3 &= 5 - 11\hat{\varphi} + 3\hat{\varphi}^2 \\
0 &= 3\hat{\varphi}^3 - 3\hat{\varphi}^2 - 11\hat{\varphi} + 5
\end{aligned}
$$

[25]:
```
coeff = [3, -3, -11, 5]
phi = np.roots(coeff)[-1]
M0 = (1 - phi) / 0.4
M1 = (phi - (phi ** 2) / 2) / 0.1
print(phi, M0, M1)
```

```
0.4261256711287521 1.4346858221781196 3.353341273312874
```

Increasing κ increases initial mispricing however reduces final mispricing. This makes sense as an increased κ will encourage abitrageurs to wait until period 1 as they anticipate mispricings to increase.

5.5 Question 6

5.5.1 Part A

If $1 - 2\psi < 0$ then the spread is increasing in ϕ, if it is > 0 then the spread in decreasing in ϕ, if it is 0 then the spread is not affected by changes in ϕ. ### Part B An increase in ϕ implies two opposing effects, that a dealer needs to post lower ask prices to entice an investor into buying the stock, and that the dealer is able to increase their ask price as they expected payoff of the asset is higher. The first effect dominates when $\psi < \frac{1}{2}$, the latter dominates when $\psi > \frac{1}{2}$. They are equal when $\psi = \frac{1}{2}$

```
[ ]:
```