Fundamental Analysis of Securities Trading

(I) Investments

Kuan-Lun WANG

National Taiwan University

Preliminary Draft: October 20, 2019

---

1E-mail: polyphonicared@gmail.com
2Home Page: https://www.csie.ntu.edu.tw/~d06922002/

Short Biodata

- Research interests:
  - time series models.
  - simulation modeling.
  - portfolio choice.
- Central themes of my application:
  - multivariate pairs trading in real time.
  - assets searching with a long-run equilibrium.
  - riskless portfolio building.
- Current work:
  - cointegration test.
  - structural change analysis.
  - the probability estimation of mean reversion.
Warning (1/6)

Accelerated Education (X)
A magic word. When there’s no time to teach from 1 to 10, teach only 1 and 10, and you can still claim that you taught from 1 to 10.

Speedrun (O)
A hard word. The intention of completing at least one simple trading flow as fast as possible. Moreover, we promise to obey some additional rules, such as describing all key ideal.

Warning (2/6)

Figure: 投資流程圖 [2]
### Warning (3/6)

![Figure: 投資流程圖 [2]](image)

Kuan-Lun WANG  
Fundamental Analysis of Securities Trading

### Warning (4/6)

![Figure: 投資流程圖 [1]](image)

Kuan-Lun WANG  
Fundamental Analysis of Securities Trading
Warning (5/6)

<table>
<thead>
<tr>
<th>Language</th>
<th>files</th>
<th>blank</th>
<th>comment</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python</td>
<td>345</td>
<td>823</td>
<td>219</td>
<td>8355</td>
</tr>
<tr>
<td>MATLAB</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>SUM:</td>
<td>346</td>
<td>824</td>
<td>224</td>
<td>8380</td>
</tr>
</tbody>
</table>

*Table: Count Lines of Code (by cloc)*

Warning (6/6)

Version of Slide

- Python 3.7.4
  - There are so many bugs in previous versions.  
- numpy '1.17.2'
- scipy '1.3.1'
- matplotlib '3.1.1'

*https://docs.python.org/3/whatsnew/changelog.html*
**Elements of Investments**

### Assets

<table>
<thead>
<tr>
<th>Real Assets</th>
<th>Financial Assets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset</td>
<td>Physical</td>
</tr>
<tr>
<td>Value</td>
<td>Substance and Properties</td>
</tr>
<tr>
<td>Example</td>
<td>Land and Buildings</td>
</tr>
</tbody>
</table>

*Table: Real Assets vs. Financial Assets*

### Financial Assets

<table>
<thead>
<tr>
<th>Debt</th>
<th>Equity</th>
<th>Derivatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Promised</td>
<td>Debt</td>
<td>Ownership</td>
</tr>
<tr>
<td>Relationship</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Financial Condition</td>
<td>Prorated</td>
<td>Other Assets</td>
</tr>
</tbody>
</table>

*Table: Type of Financial Assets*
The Investment Process

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Top-Down</th>
<th>Bottom-Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>Asset Allocation</td>
<td>Security Selection</td>
</tr>
<tr>
<td>Stable Focus</td>
<td>O</td>
<td>X</td>
</tr>
<tr>
<td>Focus</td>
<td>X</td>
<td>O</td>
</tr>
</tbody>
</table>

Table: Top-Down vs. Bottom-Up

Is there a strategy with winning percentage 100%?

Of course, it does exist.
But why do we not like it?
Financial markets must be non-efficiency or near-efficiency.

Investors have profit opportunities.

Investment philosophy: catching inefficiency in trading.

**Figure:** Return and Risk
Mean–Variance Analysis (2/2)

The Risk-Return Trade-Off [1]

- Risk-free rate: the rate of return that can be earned with certainty. e.g., Treasury bills.
- Risk premium: an expected return in excess of that on risk-free securities. e.g., index fund.
- Excess return: rate of return in excess of the risk-free rate.

That is, we involve the risk in stocks in exchange for the excess return.

NO RISK NO REWARD!!

Trading Systems and Methods

- Requirement
  - The prefect environment for IT (hardware & software)
  - The nice financial engineering (probability models)
  - the low transaction cost

- Orientations
  - Electronic trading
  - Quantitative trading (Quants)
  - High-frequency trading (maybe 0.001s)
  - Algorithmic trading
Pairs trading strategies (PTSs),
A popular short-term speculation strategy on Wall Street [4],
Two-step process:
1. Find two assets with prices that move together, historically;
2. Monitor the subsequent trading spread between the assets. Should the spread diverge, short the higher priced asset and long the lower priced asset until the spread converges again.

What are “move together” and “converges”?

(a) Log Prices Process
(b) Mean Reversion Process

Figure: An Example for PTSs
Intent: equilibrium relationships [3, 4, 5, 6].
E.g., $\beta' y_t = \beta_1 y_{1t} + \cdots + \beta_K y_{Kt} + c = 0$.

Strategies: mean-reverting strategies.
Assumption: relatively large price moves will not continue,
Cost: less slippage than trend trading strategies.

Goal: statistical arbitrage.

Now, we consider a simple bets with two outcomes.
One involving losing the 0.004 bet in probability 0.3, and the other involving winning the 0.004 bet in probability 0.7.
That is, we have
- Win: 1 $\implies 1.004$
- Loss: 1 $\implies 0.996$

Why the bets not fair?
Pairs Trading Strategies (5/6)

First
(1, X, 0, 1)

WIN
(X, 1.004, +0.004, 1) (1, 0.996, -0.004, 1.004)

LOSS
WIN
(X, 1.004, +0.004, 1) (1, 0.996, -0.004, 1.004)

The probability in the case (LOSS, LOSS, LOSS) is very low. So, the above strategy should be to win the game in high probability.

Figure: A Strategy for this Game

Kuan-Lun WANG  Fundamental Analysis of Securities Trading
Get Data

Source: TWSE (Data E-Shop)
The Data E-Shop of the TWSE provides a way for you to obtain the trading data more easily.


Source: TWSE
Taiwan Stock Exchange (TWSE) supplies daily quotes. a


---

Requirements

A. Attendance (30 hr):
   - grade_A = min{2.5A, 50}.
   - i.e., 0 ≤ grade_A ≤ 50.

B. Midterm Exam:
   - 0 ≤ grade_B ≤ 20.

C. Final Exam:
   - 0 ≤ grade_C ≤ 30.

Grade = grade_A + grade_B + grade_C

Pass if Grade ≥ 70.

Notation: Exam

Take home, and has a one on one meetings (10 min - 15 min?) in class.
Listing 1: cmd
1. `python` (open python in cmd)
2. `>> type(1)` (only display in python)
3. `>> '1'` (only display in python)
4. `>> type('1')` (only display in python)
5. `>> print('1')` (print in cmd!!)
6. `1` (print in cmd!!)

Listing 2: scripts.py
1. `print(1)`
2. `a=2`
3. `print(a)`
4. `c=1+a`
5. `print(c)`

Listing 3: cmd
1. `python scripts.py` (1 is not display, this display is print(1))
2. `2` (save 2 as a and print it)
Scripts in Python (2/2)

Listing 4: scripts.py

```python
1 a='0'
2 print(a)
3 b='K'
4 print(b)
5 c=a+b+str(1) ('str' converts 1 to str type.)
6 print(c)
```

Listing 5: cmd

```
1 python scripts.py
2 O
3 K
4 OK1
```

Functions in Python

Listing 6: addfunction.py

```python
1 def addfunction(a,b):
2    return a+b
3
4 if __name__=="__main__":
5    addfunction(1,2)
6    print(addfunction(1,2))
7    print(addfunction('0','K'))
```

Listing 7: cmd

```
1 python addfunction.py
2 3
3 3
```

Kuan-Lun WANG  Fundamental Analysis of Securities Trading
Help in Python

**Listing 8: cmd**

```python
>>> help()
>>> list
```

**Listing 9: help of list**

```python
class list (object)
    | list(iterable=(), /)
    | Built-in mutable sequence.
    | If no argument is given, the constructor creates a
```

---

**Something (1/12)**

**Listing 10: scripts.py**

```python
for n in range(3):
    print(n)
```

**Listing 11: cmd**

```python
0
1
2
```
### Listing 12: scripts.py

```
x = [n for n in range(3)]
print(x)
print(x[0:])
print(x[0:-1])
print(x[0:1])
print(x[-2:])
```

### Listing 13: cmd

```
[0, 1, 2]
[0, 1, 2]
[0, 1]
[0]
[1, 2]
```

---

### Listing 14: scripts.py

```
x = '012'
print(x)
print(x[0:])
print(x[0:-1])
print(x[0:1])
print(x[-2:])
```

### Listing 15: cmd

```
012
012
01
0
12
```
Listing 16: scripts.py

1  string='1{s}3'.format(s=2)
2  print(string)
3  print(string.replace('1','4'))
4  print(string)
5  string=string.replace('1','4')
6  print(string)

Listing 17: cmd

1  123
2  423
3  123
4  423

Listing 18: scripts.py

1  a={'s':2,3:4}
2  for key in a:
3      print(key,a[key])

Listing 19: cmd

1  s 2
2  3 4
**Listing 20: scripts.py**

```python
class tools:
    def fun(self):
        print(1)

if __name__ == '__main__':
    tools().fun()
```

**Listing 21: cmd**

```
1
```

**Listing 22: scripts.py**

```python
class tools:
    def fun(self):
        print(1)

if __name__ == '__main__':
    tools().fun()
```

**Listing 23: cmd**

```
1
```
Listing 24: scripts.py

```python
class tools():
    def __init__(self, something):
        self.hello = something
    def fun(self):
        print(self.hello)

if __name__ == '__main__':
    tools(123).fun()
```

Listing 25: cmd

```
123
```

Listing 26: scripts.py

```python
class tools0():
    def fun():
        print(1)

class tools1(tools0):
    pass

if __name__ == '__main__':
    tools1.fun()
```

Listing 27: cmd

```
1
```
Something (10/12)

Listing 28: a.py

```python
1 def fun():
2     print(1)
```

Listing 29: b.py

```python
1 import a
2 a.fun()
```

Listing 30: cmd

```
1
```

Something (11/12)

Listing 31: path

```
root
|-- folder a
|   |-- aa.py
|   |-- b.py
|   `-- main.py
```

Listing 32: a/aa.py

```python
1 def fun():
2     print(123)
```
Listing 33: b.py

```python
1 import sys
2 import a.aa as aaa
3 sys.modules['b.bb'] = aaa
```

Listing 34: main.py

```python
1 from b.bb import fun
2 fun()
```

Listing 35: cmd

```bash
1 python main.py
```

References (1/1)


