

基于Barra模型的业绩归因分析



崔诗颖

中金财富研究

2017年12月18日

基于Barra模型的业绩归因分析

目录

1 /// 流程框架

2 /// 使用方法

3 /// 因子筛选

4 /// CTA引入

一、流程框架

□ 模型目标

- 已知某一具体产品的净值与持仓数据时，对其风格因子与行业因子做归因分析，直观反应出因子的收益贡献与风险贡献。
- 模型的实际效果为：初步实现了每日自动从wind导入行情与宏观数据、自动更新因子库、自动针对每个基金产品算其收益与风险归因，并在可视化界面中看其累计情况。

一、流程框架

□ 基于多因子模型 (MFM)

$$\tilde{r}_p = \sum_{j=1}^N w_j \left(\sum_{k=1}^K x_{jk} \tilde{f}_k + \tilde{u}_j \right)$$

\tilde{r}_p : 股票总收益

x_{jk} : 因子载荷

f_k : 单位头寸时因子收益率

w_j : 股票j在组合中占的比重

$$\sigma_p = \sqrt{w^T (XFX^T + \Delta) w}$$

F : 因子收益率协方差矩阵

一、流程框架

模型重要参数估计： f_k F

- 其中 f_k 由已知个股/组合收益率序列及因子载荷，用回归方法得到。
- F 为 f_k 的协方差矩阵，可用指数加权改进。

归因结果：

- $\sum_{j=1}^N w_j x_{jk} \tilde{f}_k$ 为第 k 个因子的收益贡献

在求 f_k 时有两种处理方式：

- 时间序列回归法
- 横截面回归法

经检验，无法用时间序列回归，不能通过单位根检验，且时序存在Arch效应

一、流程框架

□ 横截面回归法

- 对于投资组合 \tilde{r}_p ，有n支股票，股票权重分配为w；k个因子，因子载荷为 x_{jk}
- 对每天的n支股票，其股票收益率为Y，k个 x_{jk} 为X，回归出的 f_k 为当日每只股票对因子K的收益率（斜率），注意 x_{jk} 为n天前
- 当日收益贡献为 $\sum_{j=1}^n x_{jk} * f_k$
- 可以辅助基金经理量化选股，规避风险
- 清晰看到每只股票的因子收益情况
- 不用rolling，直接每天取当日算，且股票有3000支，回归数据集大

一、流程框架

- a) 从wind提供的量化接口中导入沪深所有股票的所需指标，方便后续更新因子库使用（所有数据采用空值向前取）

【收盘价、开盘价、最高价、最低价、市值、成交量、收益率、换手率、一致预测净利润2年复合增长率、净资产收益率TTM、市盈率PE、Beta50、同比增长率、自由流通股本、ATR波幅、负债合计、长期负债占比、资产总计、实收资本（股本）、所属行业wind代码】存入到stock.pickle（三维表）模块中

【SZ50,HS300,ZZ500,ZZ800,ZZ1000】的收益率存入到index.csv中

```
wind_get_data.py x
1  -*- coding: utf-8 -*-
2  """
3  Created on Mon Dec 11 11:44:29 2017
4
5  @author: Rebecca Cui
6  """
7
8  import ...
9
10 #-----根据需要修改下面的参数-----
11 market_cols = ['close', 'open', 'high', 'low', 'mkt_cap_ard', 'volume', 'pct_chg',
12               'turn', 'west_netprofit_CAGR', 'roe_ttm2', 'prt_topnstocktonav',
13               'beta_100w', 'yoyeps_basic', 'free_float_shares', 'ATR',
14               'tot_non_cur_liab', 'tltoebitda', 'tot_assets', 'tangibleasset',
15               'industry_gicscode']
16
17 index_codes = ['000016.SH', '000905.SH', '000906.SH', '000852.SH', '000300.SH']
18 beginDate = '2015-06-01'
19 path = r'C:\\DELL\\internship\\CICC\\Barra\\raw data' # 结果储存路径
20 #-----
21 today = datetime.date.today().strftime('%Y-%m-%d')
22 endDate = (datetime.date.today()-datetime.timedelta(days=1)).strftime('%Y-%m-%d')
23 w.start()
24
25 def GetMarketInfo(code, cols):...
26
27 def Align(code, df):...
28
29 def Concat(codes, cols):...
30
31 # 通过wset取当前沪深300成份股
32 stockSector = w.wset("sectorconstituent", "date="+today+",sectorid=1000000090000000")
33
34 dates, codes, names = stockSector.Data
```

一、流程框架

b) 建立全股票池的因子库，目前共纳入因子31个，因子文件存入Database中，在这一步不做标准化、去极值等任何预处理，同时更新中信行业因子

```
9
10 #property copyright "Shiying Cui"
11 #property version "5.0"
12 #property private!!!
13 #####1 收盘 2 开盘 3 最高 4 最低
14 #####9 一致预测净利润2年复合增长率 10 净资产收益
15 #####13 同比增长率 14 自由流通股本 15 ATR波动
16
17 class style_fac(object):
18     def __init__(self, data, index):
19         self.data = data
20         self.index = index
21         ##存各指数价格，上证指数为0，沪深300为1，中证5
22
23     ###1
24     + def bigsize(self):...
25
26     ###2
27     + def medsize(self):...
28
29     ###3
30     + def high_low(self):...
31
32     ###4
33     + def retv(self):...
34
35     ###5
36     + def turn(self):...
37
38     ###6
39     + def wgt_rt(self):...
40
41     ###7
42     + def halpha(self):...
```

```
129     ###8
130     + def vol(self):...
131
132     ###9
133     + def CAGR(self):...
134
135     ###10
136     + def ROE(self):...
137
138     ###11
139     + def topstock(self):...
140
141     ###12
142     + def increase(self):...
143
144     ###13
145     + def EY(self):...
146
147     ###14
148     + def MLEV(self):...
149
150     ###15
151     + def DTOA(self):...
152
153     ###16
154     + def BLEV(self):...
155
156     ###17
157     + def B300(self):...
158
159     ###18
160     + def B500(self):...
161
162     ###19
163     + def B800(self):...
164
165     ###20
166     + def B1000(self):...
167
168     ###21
169     + def R300(self):...
```

```
352     ###22
353     + def R500(self):...
354
355     ###23
356     + def R800(self):...
357
358     ###24
359     + def R1000(self):...
360
361     ###25
362     + def ASI(self):...
363
364     ###26
365     + def DDI(self):...
366
367     ###27
368     + def Hurst(self):...
369
370     ###28
371     + #####P1, P2默认值为3#####
372     + def KDJ(self, P1, P2):...
373
374     ###29
375     + def MFI(self):...
376
377     ###30
378     + def Ulcer(self):...
379
380     ###31
381     + def BR(self):...
382
383     ###industry
384     + def industry(self):...
385
386     + ##将各股收益率存入表中
387     + def netrt(self):
388         temp = self.data.iloc[:, :, 6]
389         temp.to_csv(r"C:\\Dell\\internship\\
```


一、流程框架

c) 处理每日基金底层持仓数据

修正错误：

观察到基金底层持仓表中股票组合（1102）占净值百分比，因此需要从基金中提取股票组合的权重及股票组合的累计净值。

此前对于基金收益率直接拿基金累计净值计算得来【错】

表中：

1102代表股票投资

SH 代码为 0101

估值增值为0199 0701

SZ 代码为 3101，4101

估值增值为 3199，4199，3301

估值增值和成本股等市价为0，都未计入持仓中。

证券投资基金估值表

恒生电子____中金浦泰艾方量化金选1号私募投资基金____专用

估值日期：2017-03-07

单位净值：1.005

科目代码	科目名称	市值占净值%	估值增值
102113	期货清算备付金	3.8369	
10211301	永安期货	3.8369	
1031	存出保证金	0.6002	
103106	券商保证金-中金公司	0.1719	
103113	期货交易存出保证金	0.4283	
10311322	永安期货	0.4283	
1102	股票投资	22.8285	320,642.47
110201	上交所A股	11.0385	108,701.89
11020101	上交所A股成本	11.0385	108,701.89
11020199	上交所A股估值增值	0.1162	
110231	深交所A股	10.0409	177,665.63
11023101	深交所A股成本	10.0409	177,665.63
11023199	深交所A股估值增值	0.1900	
110241	深交所A股_创业板	1.7491	34,274.95
11024101	深交所A股成本_创业板	1.7491	34,274.95
11024199	深交所A股估值增值_创业板	0.0367	
1105	基金投资	64.3161	4,810.89
110503	ETF基金	64.3161	4,810.89
11050301	ETF基金成本	64.3161	4,810.89
11050301511990	华宝添益	64.3161	4,810.89
11050399	ETF基金估值增值	0.0051	
11202	买入返售金融资产	8.5555	

一、流程框架

c) 处理每日基金底层持仓数据

算法流程：

对每日底层持仓数据表，提取其基金名称，并通过字符串切割提取代码将其转换为wind代码

该基金所有持仓数量放在一张csv中，未出现的股票直接赋值为0

除此外，根据1102股票组合占净值百分比，算出纯股票组合的产品净值，在此基础上算出收益率，作为接下来回归的y

```
10 path = r'C:\DELL\internship\CICC\Barra\test'
11 file_list = os.listdir(path) # 指定文件夹中的所有文件
12 # 删除文件夹中无用的表
13
14 def Replace(s):
15     dic = {'0101': '.SH', '01990': '.SH',
16           '3101': '.SZ', '4101': '.SZ', '31990': '.SZ', '41990': '.SZ'}
17     prefix = s[-6:]
18     suffix = s[4:8]
19     ss = prefix + dic[suffix]
20     return ss
21
22
23 def GetEquity(df):...
24
25
26
27 def GetValue(df):...
28
29
30
31 #ser = GetNet(df)
32 rt_equity = pd.DataFrame()
33 rt_cols = []
34 rt_vals = []
35 pt_equity = pd.DataFrame()
36 pt_cols = []
37 pt_vals = []
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52 for file in file_list:
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72 rt_cal = pd.DataFrame(rt_vals, index=rt_cols, columns=['equity'])
```

一、流程框架

d) 收益归因分析

对所有参与因子做标准化与去极值处理（标准化保证均值为0）

对基本面因子取窗口为8天，技术面因子取窗口为2天

做不带截距项的加权最小二乘回归，加权方法参考于广发研报（市值的加权）

不做参数优化，不设置in-out sample

输出结果为3个，分别是各因子收益贡献绝对值、各因子收益占比，各因子

```
1 import numpy as np
2 import pandas as pd
3 import datetime as dt
4 from datetime import timedelta
5 import statsmodels.api as sm
6 import scipy.stats.mstats as ssm
7
8 class regress(object):
9     def __init__(self, today, data, hold, tot_ret):...
16
17     def date_win(self, win1, win2):...
33
34     def pre(self, low_limit, up_limit):...
166
167     def industry(self):...
177
178     def factor_ret(self):...
332
```

参数：

Today为当前希望分析的日期；

Data为更新的行情库

Hold为处理好的增量持仓数据表

Tot_ret为处理好的股票部分收益率

Win1取2天，win2取8天

Low_limit和up_limit 取0.01，主要是刨去inf值，标准化后会变成+、- inf

一、流程框架

e) 风险归因分析

对于风险协方差：

$$F_{kd}^{(d)} = cov(f_k, f_l) = \sum_{s=t-h}^t \tau^{t-s} (f_{ks} - f_{kmean})(f_{ls} - f_{lmean}) / \sum_{s=t-h}^t \tau^{t-s}$$

边际风险贡献FMCAR：

$$FMCAR = \frac{F * X_{pA}}{\sigma_p}$$

参数：

ito为半衰期，意义为第t-ito/2天的权重为第t天的一半；

win为窗口，即时间序列长度

总风险贡献即为风险暴露乘边际风险贡献

输出结果为风险贡献和风险贡献占比

Win取40，ito取5

不做参数优化，不设置in-out sample

二、使用方法

__pycache__	BarraSystemV10: python project 打包所有程序
BarraSystemV10	Database: 风格与行业因子数据库
Database	Raw data: 存放股票与股指行情数据、基金股票部
Presentation	分累计净值、收益率、基金持仓数据
raw data	
test	Test: 存放所有的基金底部持仓数据（日更）

» CICC » Barra » BarraSystemV10

名称

__pycache__
backtest.py
factordata.py
prototype.py
PrototypeMain.py
recursion.py
regress.py
wind_get_data.py
xw_test.py

Backtest: 因子回测文件（定义具体函数类）

Factordata: 更新每日因子库的子类

Prototype: py GUI 框架参数与函数定义（父类）

PrototypeMain: 执行函数【每日运行该程序即可弹出图形界面，方便操作】

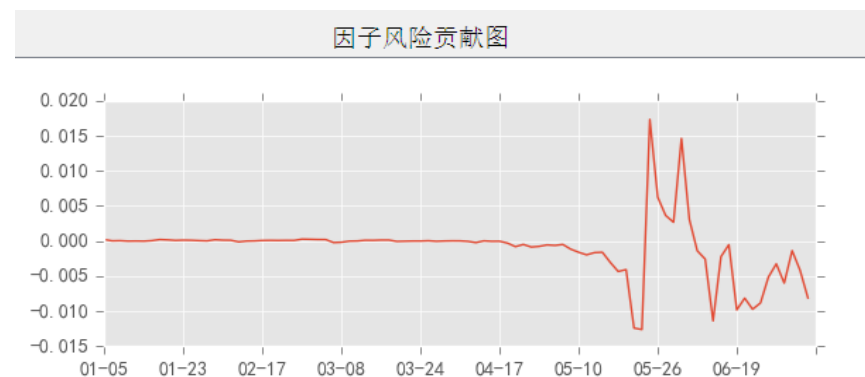
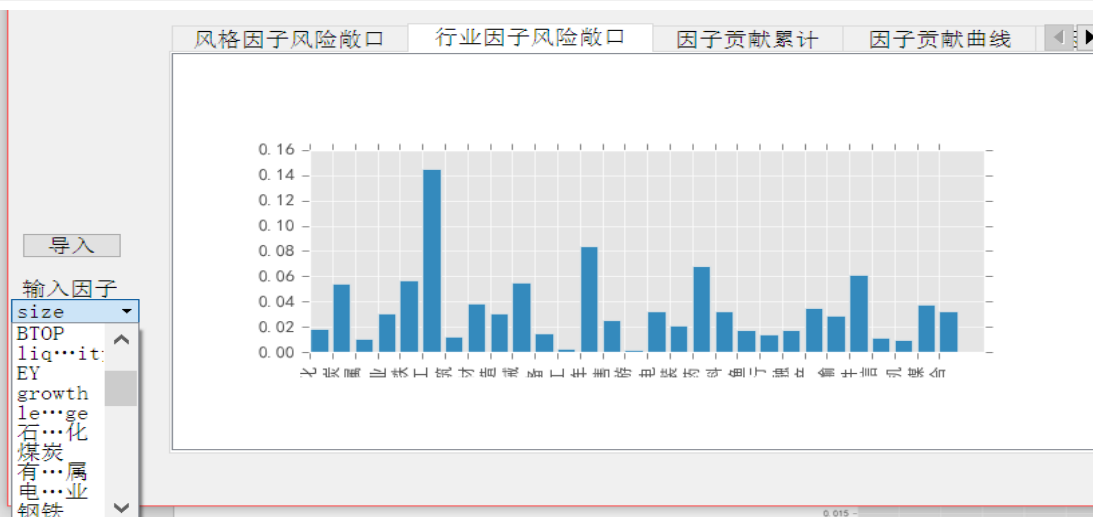
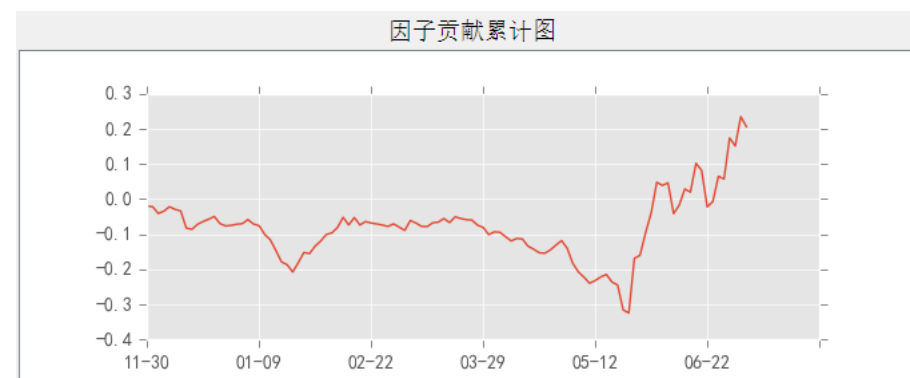
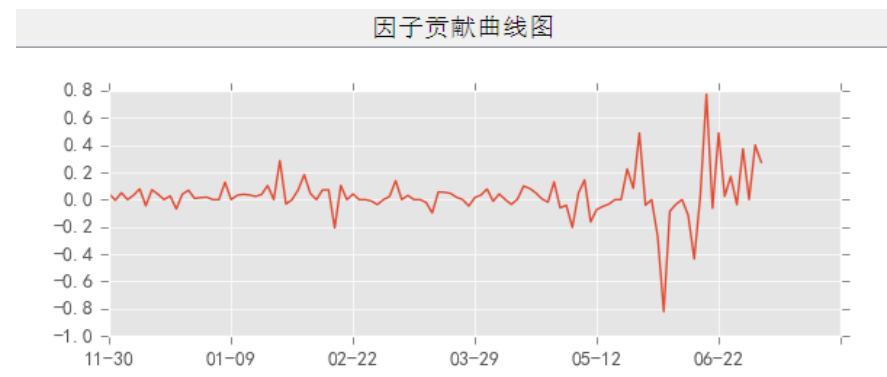
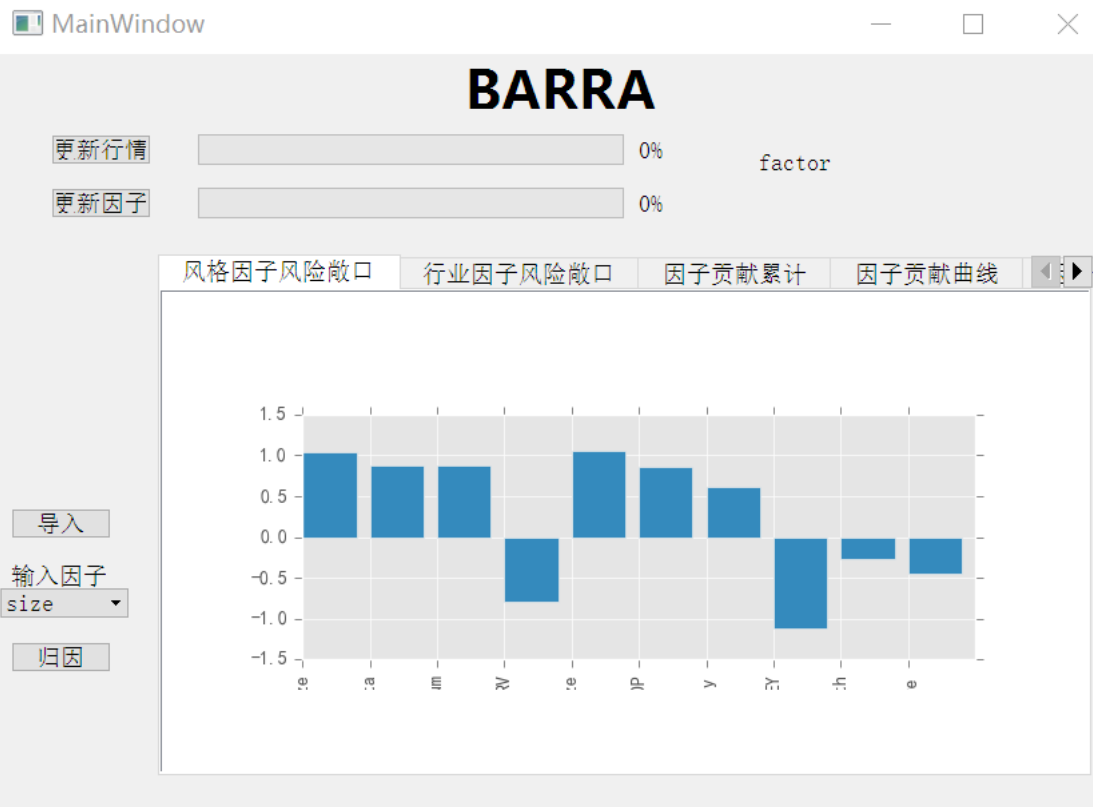
Recursion: 对因子进行回测，输出十几个指标，可每月运行一次，找出该月表现较好的因子进行归因

Regress: 两个类：收益归因与风险归因

Wind_get_data: 从wind量化接口日更所需数据：stock和stock_index两张表

Xw_test: 处理每日基金持仓底表，更新各基金的hold(持仓表)和net_value(股票部分净值表)

二、使用方法



三、因子筛选

- Beta
- EY
- Growth
- industry
- Leverage
- Momentum
- Residual
- Return
- Size
- TA
- Tech
- Turn
- value
- Volatility

因子目标

- 1) 避免大量公式复杂的价量因子，选择直观与逻辑上便于理解的
-----此处为与多因子选股不同之处
- 2) 因子间要有低相关性：
-----构造了一个衡量相关性强弱的构造指标（针对月度选择因子）
- 3) 因子回测文件 ----recursion.py 执行文件
----backtest.py 函数脚本
- 4) 因子筛选：
月度调仓，人工手动分析，观察各因子IC和IC decay

三、因子筛选

大类因子	小类因子	因子计算方式
Size	LNCAP	个股总市值对数：反应大盘股。
Beta	BETA(50,300,500,800,1000)	$r_i = \alpha + \beta r_m + e_i$ ：利用个股收益率序列和沪深300指数收益率序列进行一元线性回归，收益率序列长度取252交易日。个股收益率序列和沪深300指数收益率序列均以半衰指数加权，半衰期为63日。 [表现最好的为800]
RV	R(50,300,500,800,1000)	上述回归中的残差序列，由于选定了800，因此残差也选800。
Residual Volatility	HAlpha	个股500天收益与上证综指回归的截距项。
	Wgt_rt	个股1个月以每日换手率作为权重对每日收益率求算数平均。
Non-linear Size	NLSIZE	SIZE的立方：反应中盘股。

三、因子筛选

大类因子	小类因子	因子计算方式
TA	vol	过去一个月成交量的标准差。
Liquidity	STOM	$STOM = \ln(\sum_{t=1}^{21} \frac{V_t}{S_t})$ ，其中 V_t 表示当日成交量， S_t 表示流通股本。
	STOQ	$STOQ = \ln(\frac{1}{T} \sum_{\tau}^T \exp(STOM_{\tau}))$ ：其中 $T = 3$ ，月度。
	STOA	$STOA = \ln(\frac{1}{T} \sum_{\tau}^T \exp(STOM_{\tau}))$ ：其中 $T = 12$ ，月度。
Volatility	hml	过去1个月每日最高价-最低价序列的标准差。
	retv	过去1个月收益率的标准差。
Growth	CAGR	一致预测净利润2年复合增长率。
	increase	企业盈利同比增长率。
	ROE	净资产收益率TTM。

三、因子筛选

大类因子	小类因子	因子计算方式
Leverage	MLEV	$MLEV = (ME + LD) / ME$ ：其中ME表示企业当前总市值，LD表示企业长期负债。
	DTOA	$DTOA = TD / TA$ ：其中TD表示总负债，TA表示总资产。
	BLEV	$BLEV = (BE + LD) / BE$ ：其中BE表示企业账面权益，LD表示企业长期负债。
Turn	turn	过去一个月换手率的标准差。
Value	EP	市盈率ttm的倒数
Tech	ASI	累计振动升降指标。
	BR	多空双方力量强弱指标：1月中h-c的和除以c-l的和。
	DDI	方向标准离差指数。
	Hurst	Hurst条件概率指标。
	KDJ	KDJ随机指标。
	MFI	资金流量指标。
	Ulcer	收盘价相对于最高价变动率指标。

三、因子筛选

判断因子相关性

$$C_{AB} = \frac{\text{mean}(\text{Corr}_i^{AB} | i = 1, \dots, N)}{\text{std}(\text{Corr}_i^{AB} | i = 1, \dots, N)}$$

该指标逻辑如下：

Corr_i^{AB} 为A与B两个因子在第 i 天对全股票池横截面的相关系数

按照更新归因模型的频率来看（每月更一次），N可以取20(即20个交易日)

若想知道两个重要因子间是否长期相关，可以将N取久一些，例如取600.

【注：该指标为自己构造，可能不准确，慎用】

四、CTA引入

目标：判断市场处于震荡还是趋势

涉及领域：小波去噪、信号处理

两步法：

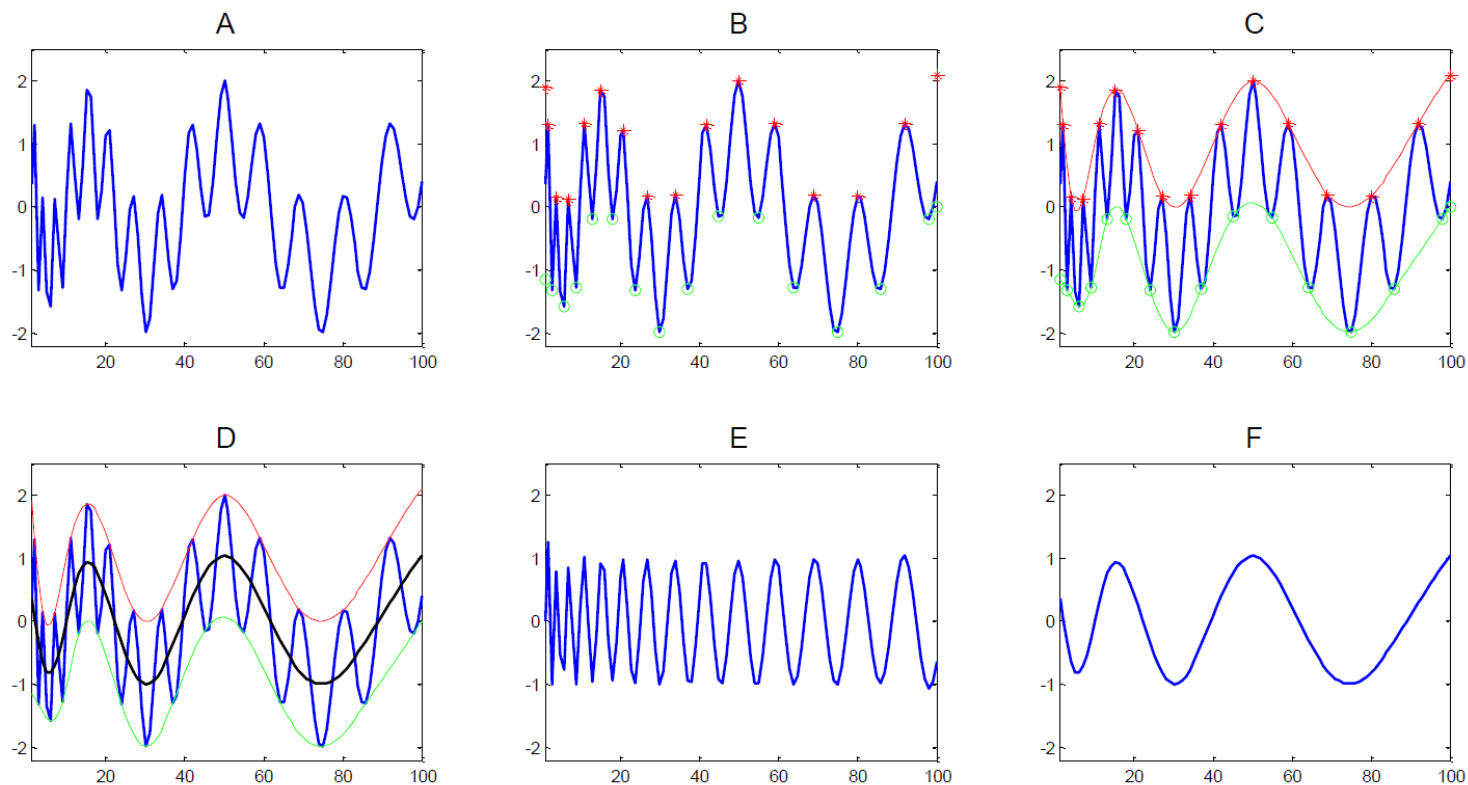
- 1. 运用历史数据优化出好的时点**
- 2. 运用机器学习找出好的时间间隔进行预测**

四、CTA引入

EMD经验模态分解

复杂信号分解为 波动项+趋势项：

$$s(t) = \sum_{i=1,2,\dots,n} IMF_i(t) + r_n(t)$$



四、CTA引入

R-Breaker实证分析

右图为主函数

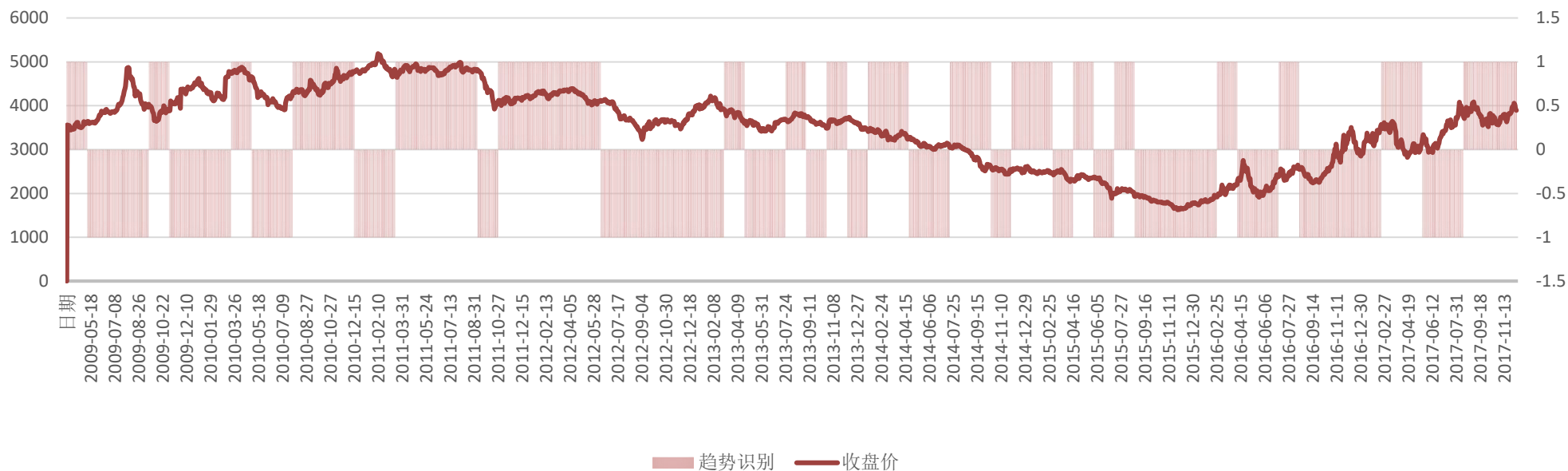
按比例止盈止损
初步定为止盈10%
止损5%

```
57 def run(self):
58     self.counter = self.counter + 1
59     if self.counter >= self.data_freq / self.Run_freq:
60         self.cal_thr()
61         self.counter = 0
62     if self.ObserverSellThr is None:
63         return
64
65     trade_flag = 0
66
67     last_price = self.PriceList[-1]
68     if marketPosition.marketPosition[self.symbol] == 0:
69         if last_price < self.BreakSellThr:
70             trade_flag = -1
71             #order.insert_openShortPosition(symbol=self.symbol, volume=self.trade_shares, price=None, is_market=True)
72             #空仓情况下，盘中价格跌破突破卖出价，采取趋势策略，即在该点做空；
73         if last_price > self.BreakBuyThr:
74             trade_flag = 1
75             #order.insert_openLongPosition(symbol=self.symbol, volume=self.trade_shares, price=None, is_market=True)
76             #空仓情况下，盘中价格超过突破买入价，采取趋势策略，即在该点做多
77
78
79     if last_price > self.ObserverSellThr:
80         self.isCrossObserverSellThr = True
81     if last_price < self.ObserverBuyThr:
82         self.isCrossObserverBuyThr = True
```

四、CTA引入

结果示例：螺纹钢（目前做了21个商品品种和南华指数）

螺纹钢





基于Barra模型的业绩归因分析

财富研究部 | 2017年11月3日