法律声明

本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

关注 小象学院

第一课

期货趋势型策略开发和股票突破信号系统编写

系统化构建量化交易体系：

模块1：动手开发期货和股票量化交易策略

内容介绍

- 课程介绍
- 常用的量化平台概述
- 交易系统的核心要素
- 趋势型策略原理
- 用TB开发商品期货趋势型策略
- 用Python接入Tushare数据源来编写股票突破信号系统

系统化构建量化交易体系

课程介绍

三个模块

- 动手开发期货和股票量化交易策略
  - 交易系统核心要素及多平台下的实例编写
- 搭建自己的股票回测及交易平台
  - 从零实现一个自主可控的模块化交易系统
- 量化交易策略逻辑的深入讨论
  - 策略改进方法、实战要点、上实盘的细节

预期收获

- 从数据、模型、回测、模拟、实盘全过程的量化系统学习
- 利用多种第三方工具平台开发策略
- 对接开源数据接口开发策略
- 搭建自己的回测和交易平台
- 掌握主观策略的系统性量化方法
- 掌握策略修正优化的思路和方法
- 获得量化交易中各种实战技巧
- 获取课件、源码、量化交易系统数据库方案

自己动手做量化的起点

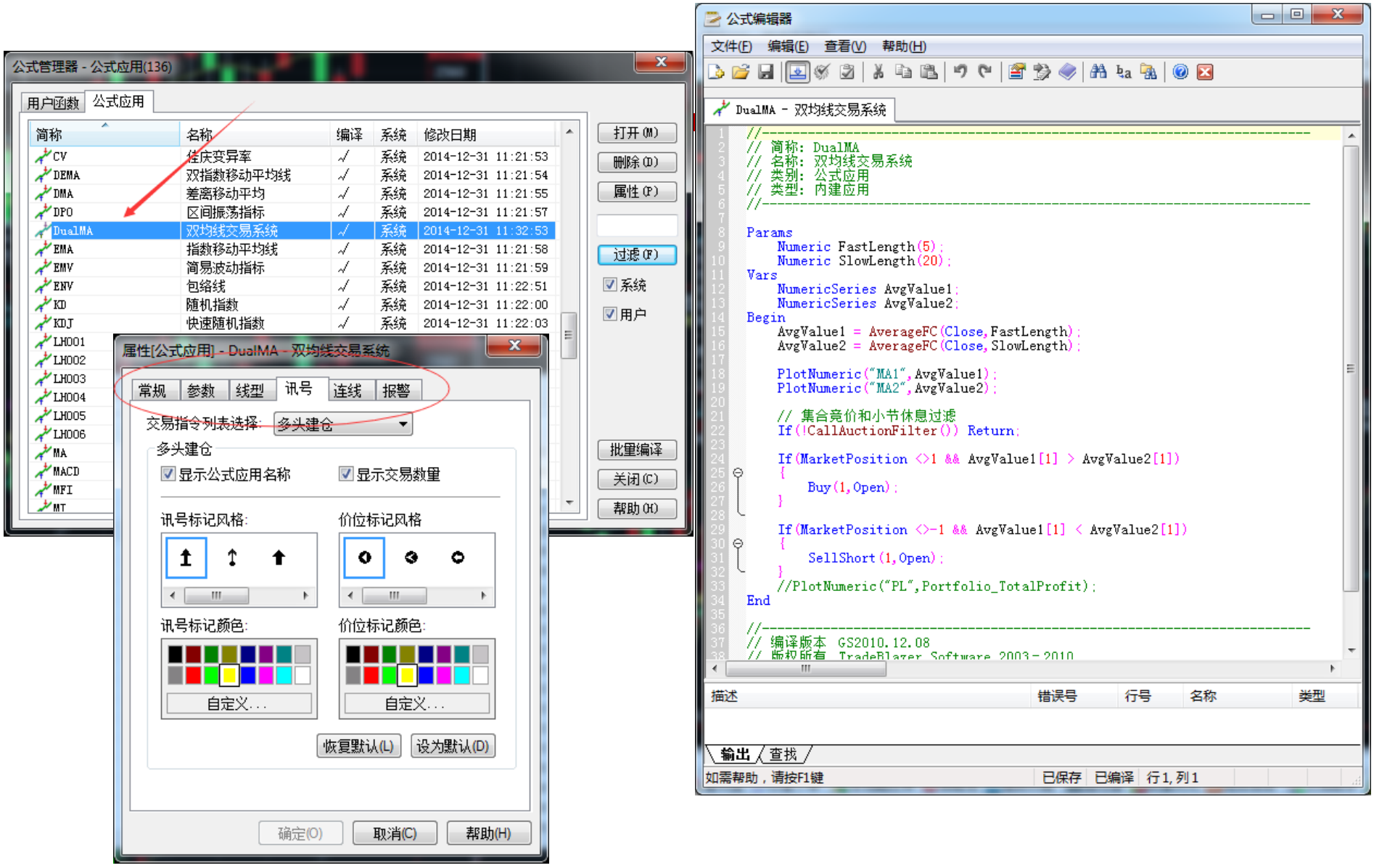
常用的量化平台概述

一个量化策略的进阶之路

第三方量化平台分类和举例

- 本地（图表/后台交易）
  - 金字塔、MC、TB、WH、TS、MT4、…
- 云端（SaaS/券商定制）
  - 聚宽、优矿、米筐、…
- SDK/量化API（+终端/Web UI）
  - 万得、东财Choice、掘金量化、…
- 开源框架（基于Python）
  - PyCTP、VNPY、QuickLib、Zipline、…


交易开拓者平台(旗舰版) 64位 - [(设置示例)双均线多周期测试-rb000 15分钟线]
文件(F) 视图(V) 插入(I) 格式(O) 周期(C) 绘图(D) 工具(M) 交易(T) 资讯(I) 窗口(W) 帮助(H) 技术支持(S)
功能模块
行情报价
分时图
超级图表
期权报价
监控器
批量优化
组合报告
高级功能
TB公式
帮助与链接
简称
白银指数
黄金指数
沥青指数
沪铝指数
螺纹钢指数
沪镍指数
沪铅指数
橡胶指数
沪铜指数
沪锡指数
线材指数
沪锌指数
热轧卷板指数
PVC指数
豆一指数
豆粕指数
豆油指数
鸡蛋指数
焦煤指数
焦炭指数
胶合板指数
聚丙烯指数
塑料指数
自选
rb000 螺纹钢指数 15分钟线 2016/08/19 21:30 C=2560 13 0.51% O=2557 H=2563..
rb000 螺纹钢指数 30分钟线 2016/08/19 22:30 C=2565 18 0.71% O=2556 H=2569..
rb000 螺纹钢指数 1小时线 2016/08/19 22:00 C=2565 18 0.71% O=2576 H=2580..
rb000 螺纹钢指数 1日线 2016/08/22 C=2565 4 0.16% O=2561 H=2580 L=2550
DualMA[1]
08月19日
08月12日
08月08日
08月11日
08月16日
08月18日
(设置示例)双均线多周期测试
闪电下单 交易师 批量下单 套利宝 价差下单 本地触发单 策略易 高级功能 监控器 批量优化 组合报告 交易助手 允许自动 交易帐户 系统设置


公式管理器 - 公式应用(136)
用户函数
公式应用
简称
名称
编译
系统
修改日期
CV 佳庆变异率 √ 系统 2014-12-31 11:21:53
DEMA 双指数移动平均线 √ 系统 2014-12-31 11:21:54
DMA 差离移动平均 √ 系统 2014-12-31 11:21:55
DPO 区间振荡指标 √ 系统 2014-12-31 11:21:57
DualMA 双均线交易系统 √ 系统 2014-12-31 11:32:53
EMA 指数移动平均线 √ 系统 2014-12-31 11:21:58
EMV 简易波动指标 √ 系统 2014-12-31 11:21:59
ENV 包络线 √ 系统 2014-12-31 11:22:51
KD 随机指数 √ 系统 2014-12-31 11:22:00
KDJ 快速随机指数 √ 系统 2014-12-31 11:22:03
LH001
LH002
LH003
LH004
LH005
LH006
MA
MACD
MFI
MT
打开(O)
删除(D)
属性(P)
过滤(F)
系统
用户
批量编译
关闭(C)
帮助(H)
属性[公式应用] - DualMA - 双均线交易系统
常规
参数
线型
讯号
连线
报警
交易指令列表选择:
多头建仓
多头建仓
显示公式应用名称
显示交易数量
讯号标记风格:
价位标记风格
讯号标记颜色:
价位标记颜色:
自定义...
自定义...
恢复默认(L)
设为默认(D)
确定(O)
取消(C)
帮助(H)
公式编辑器
文件(F)
编辑(E)
查看(V)
帮助(H)
DualMA - 双均线交易系统
// 简称: DualMA
// 名称: 双均线交易系统
// 类别: 公式应用
// 类型: 内建应用
Params
Numeric FastLength(5);
Numeric SlowLength(20);
Vars
NumericSeries AvgValue1;
NumericSeries AvgValue2;
Begin
AvgValue1 = AverageFC(Close,FastLength);
AvgValue2 = AverageFC(Close,SlowLength);
PlotNumeric("MA1",AvgValue1);
PlotNumeric("MA2",AvgValue2);
// 集合竞价和小节休息过滤
If(!CallAuctionFilter()) Return;
If(MarketPosition <>1 && AvgValue1[1] > AvgValue2[1])
Buy(1,Open);
If(MarketPosition <>-1 && AvgValue1[1] < AvgValue2[1])
SellShort(1,Open);
//PlotNumeric("PL",Portfolio_TotalProfit);
End
// 编译版本 GS2010.12.08
// 版权所有 TradeBlazer Software 2003-2010
描述
错误号
行号
名称
类型
输出
查找
如需帮助，请按F1键
已保存
已编译
行 1, 列 1

交易策略测试报告[rb000 - 15分钟线]

交易汇总 | 交易分析 | 交易记录 | 平仓分析 | 阶段总结 | 资产变化 | 图表分析 | 系统设置

性能概要

统计指标	全部交易	多头	空头
净利润	(4660.00)	(7400.00)	2740.00
总盈利	79200.00	36360.00	42840.00
总亏损	(83860.00)	(43760.00)	(40100.00)
总盈利/总亏损	0.94	0.83	1.07
交易手数	958	479	479
盈利比率	30.58%	27.77%	33.40%
盈利手数	293		
亏损手数	642		
持平手数	23		
平均利润	(4.86)		
平均盈利	270.31		
平均亏损	(130.62)		
平均盈利/平均亏损	2.07		
最大盈利	2430.00		
最大亏损	(980.00)		
最大盈利/总盈利	0.03		
最大亏损/总亏损	0.01		
净利润/最大亏损	4.76		
最大连续盈利手数	4		
最大连续亏损手数	13		
平均持仓周期	15		
平均盈利周期	30		
平均亏损周期	9		
平均持平周期	16		

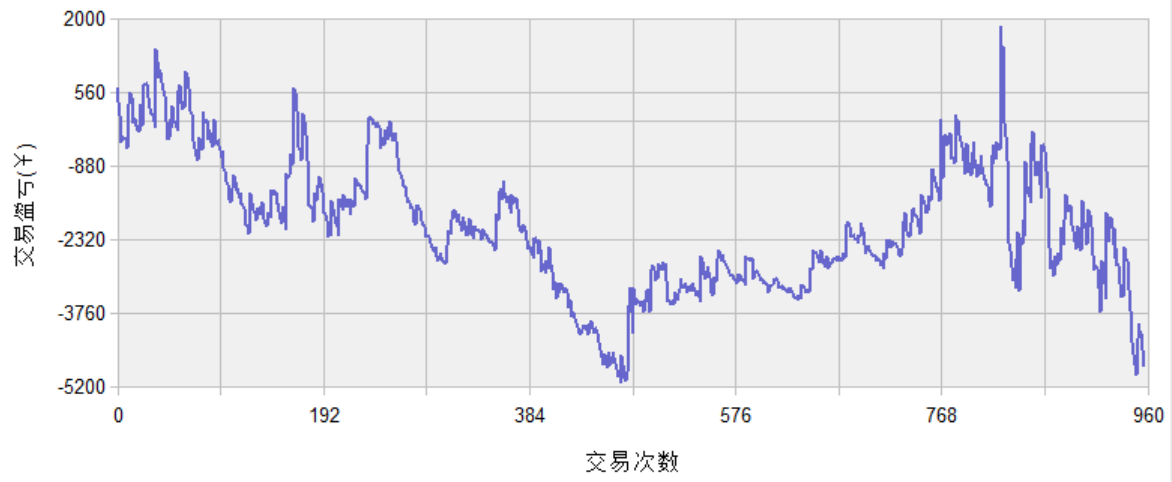
商品[1]个 公式应用[1]个

交易策略测试报告[rb000 - 15分钟线]

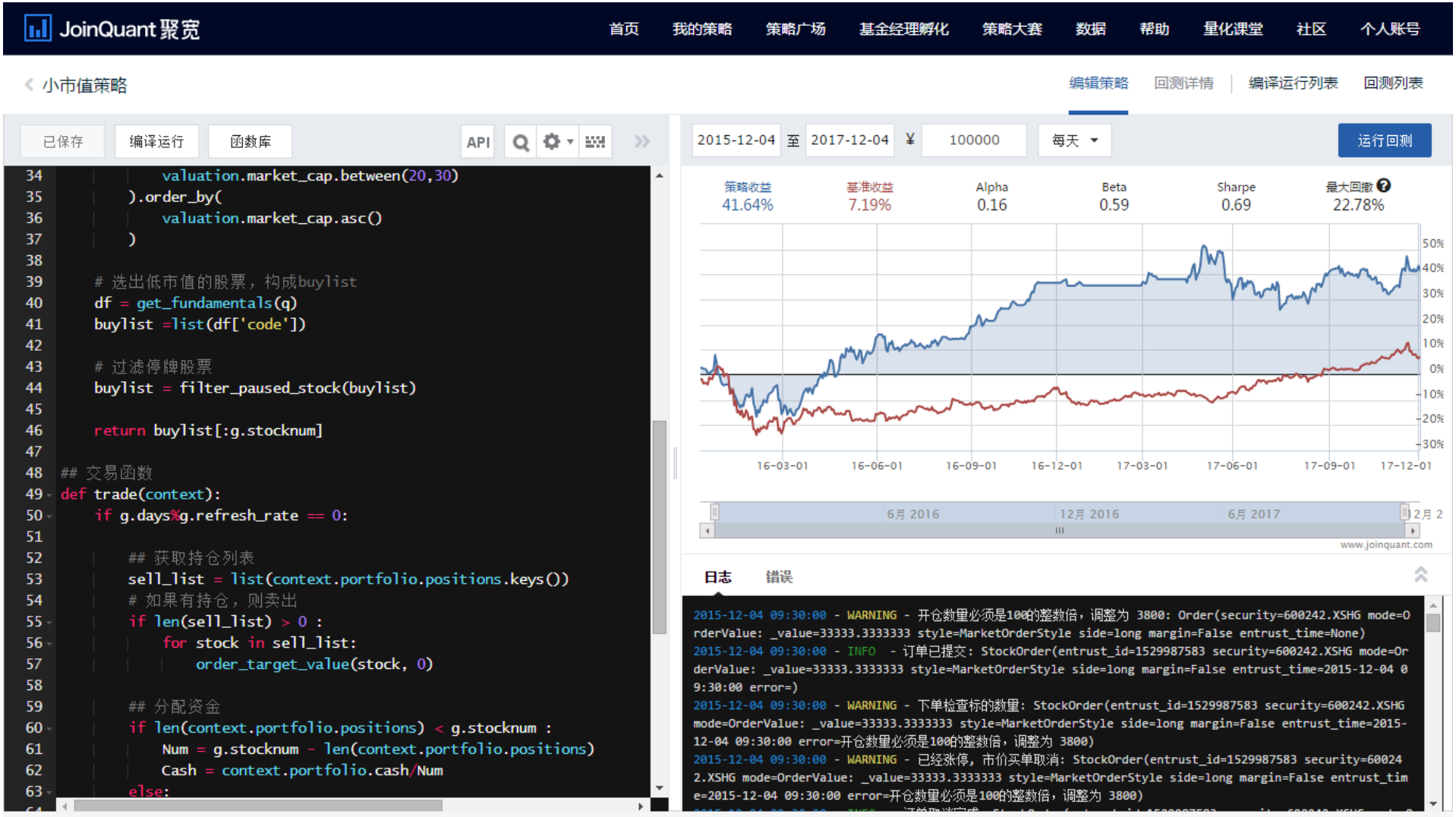
交易汇总 | 交易分析 | 交易记录 | 平仓分析 | 阶段总结 | 资产变化 | 图表分析 | 系统设置

性能图表 交易图表

交易盈亏曲线图



商品[1]个 公式应用[1]个 时间范围[2014/01/02 09:00 - 2016/08/19 22:45]


JoinQuant聚宽
首页
我的策略
策略广场
基金经理孵化
策略大赛
数据
帮助
量化课堂
社区
个人账号
小市值策略
编辑策略
回测详情
编译运行列表
回测列表
已保存
编译运行
函数库
API
2015-12-04
至
2017-12-04
¥
100000
每天
运行回测
策略收益
41.64%
基准收益
7.19%
Alpha
0.16
Beta
0.59
Sharpe
0.69
最大回撤
22.78%
50%
40%
30%
20%
10%
0%
-10%
-20%
-30%
16-03-01
16-06-01
16-09-01
16-12-01
17-03-01
17-06-01
17-09-01
17-12-01
6月 2016
12月 2016
6月 2017
www.joinquant.com
日志
错误
valuation.market_cap.between(20,30)
).order_by(
valuation.market_cap.asc()
)
# 选出低市值的股票，构成buylist
df = get_fundamentals(q)
buylist =list(df['code'])
# 过滤停牌股票
buylist = filter_paused_stock(buylist)
return buylist[:g.stocknum]
交易函数
def trade(context):
if g.days%g.refresh_rate == 0:
获取持仓列表
sell_list = list(context.portfolio.positions.keys())
# 如果有持仓，则卖出
if len(sell_list) > 0 :
for stock in sell_list:
order_target_value(stock, 0)
分配资金
if len(context.portfolio.positions) < g.stocknum :
Num = g.stocknum - len(context.portfolio.positions)
Cash = context.portfolio.cash/Num
else:
2015-12-04 09:30:00 - WARNING - 开仓数量必须是100的整数倍，调整为 3800: Order(security=600242.XSHG mode=OrderValue: _value=33333.3333333 style=MarketOrderStyle side=long margin=False entrust_time=None)
2015-12-04 09:30:00 - INFO - 订单已提交: StockOrder(entrust_id=1529987583 security=600242.XSHG mode=OrderValue: _value=33333.3333333 style=MarketOrderStyle side=long margin=False entrust_time=2015-12-04 09:30:00 error=)
2015-12-04 09:30:00 - WARNING - 下单检查标的数量: StockOrder(entrust_id=1529987583 security=600242.XSHG mode=OrderValue: _value=33333.3333333 style=MarketOrderStyle side=long margin=False entrust_time=2015-12-04 09:30:00 error=开仓数量必须是100的整数倍，调整为 3800)
2015-12-04 09:30:00 - WARNING - 已经涨停，市价买单取消: StockOrder(entrust_id=1529987583 security=600242.XSHG mode=OrderValue: _value=33333.3333333 style=MarketOrderStyle side=long margin=False entrust_time=2015-12-04 09:30:00 error=开仓数量必须是100的整数倍，调整为 3800)


JoinQuant 聚宽
首页
我的策略
策略广场
基金经理孵化
策略大赛
数据
帮助
量化课堂
社区
个人账号
小市值策略
编辑策略
回测详情
编译运行列表
回测列表
设置： 2015-12-04 到 2017-12-04， ¥100000， 每天
状态： 回测完成，耗时10.42s
模拟交易
归因分析
分享到社区
查看代码
导出
收益概述
交易详情
每日持仓&收益
日志输出
性能分析
策略收益
基准收益
Alpha
Beta
Sharpe
Sortino
Information Ratio
Volatility
Benchmark Volatility
收益概述
策略收益 41.64%
策略年化收益 19.48%
基准收益 7.19%
Alpha 0.157
Beta 0.588
Sharpe 0.693
胜率 0.590
盈亏比 1.499
最大回撤 22.779%
其他指标
缩放： 1个月 1年 全部
策略收益
基准收益
超额收益
普通轴
对数轴
超额收益
时间： 2015-12-04 - 2017-12-04
50%
40%
30%
20%
10%
0%
-10%
-20%
-30%
收益
0k
-10k
-20k
每日盈亏
100k

三大类接口

数据接口

支持股票、指数、期货、期权、债券、基金等各种标的，提供实时行情，历史行情，财务数据、EDB宏观经济数据

交易接口

为用户提供实盘交易接口，现已对接国内数十家券商的交易柜台，同时，还是支持大部分期货公司的CTP交易接口，为用户实现程序化交易提供方便。

支持的券商列表

工具接口

通过接口，可以实现对PMS组合管理、WTTS模拟交易的程序化操作。

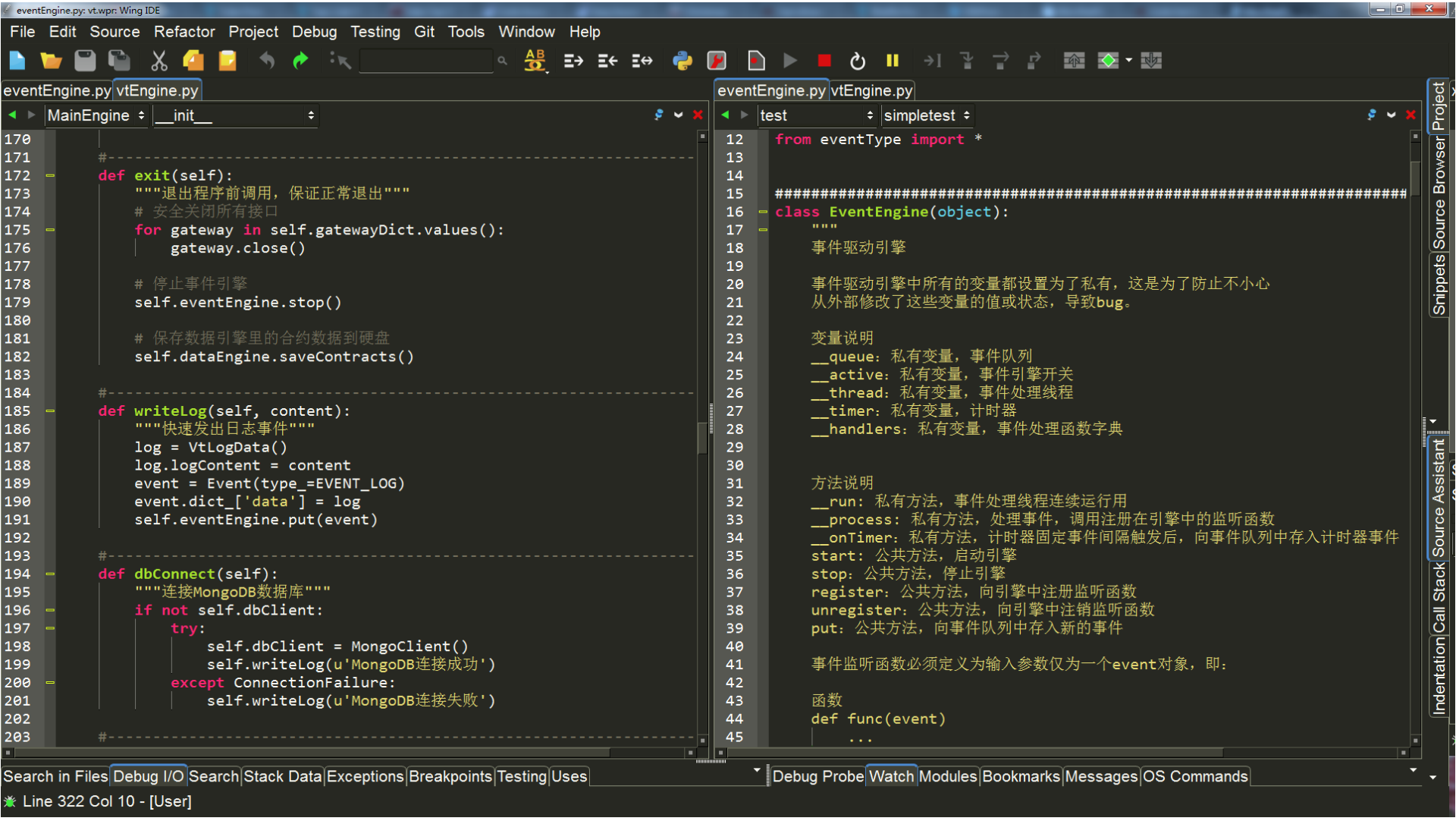
如何使用PMS进行回测

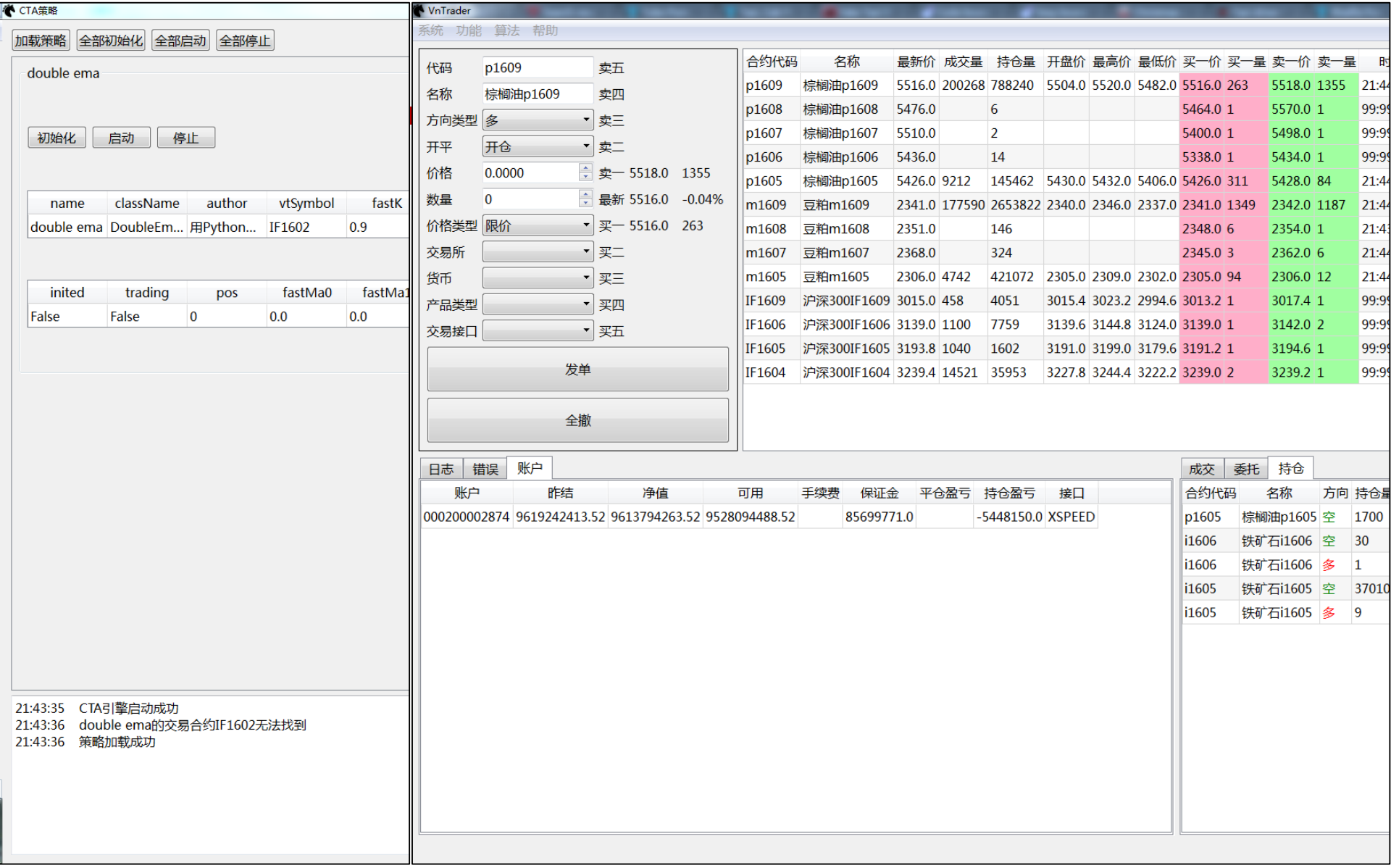
```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

from WindPy import w
from pymongo import MongoClient

from datetime import *
import sys
import random

mongoUrl = 'mongodb://127.
client = MongoClient(mongoU
db = client.quant
coll_sec = db.securities
coll_trd = db.trades

# Read out all available s
cursor = coll_sec.find({},
stock_list = [obj["trade_c

# Start WindPy module
w.start()
if not w.isconnected():
    print("Error connecting
    sys.exit(-1)

# Get all valid trading da
result = w.tdays("2015-01-
if result.ErrorCode != 0 or
    print("Error to get tra
    sys.exit(-1)
trade_dates = [dt.strftime
```

```python
for dt in trade_dates[::3]:
    print(dt)
    if dt == start_date:
        # Buy-in first bucket of stocks
        cursor = coll_trd.find({"code":{"$in":stock_pool}, "date":dt, "trade_status":u"交易"},
        records = [rec for rec in cursor]
        count = len(records)

        for rec in records:
            obj = OneStock()
            obj.trade_code = rec["code"]
            obj.init_price = obj.last_price = rec["open"]
            obj.amount = int(capital_in_position / count / rec["open"] / 100) * 100  # Evenly
            account.append(obj)

        # Issue bucket buy-in request
        codes = ",".join([obj.trade_code for obj in account])
        amts = ",".join([str(obj.amount) for obj in account])
        result = w.bktorder("%s 9:45:00" % dt, codes, "Buy", amts, 'Price=Open')
        if result.ErrorCode != 0:
            print("Error putting buying orders, aborted!")
            sys.exit(-1)
    else:
        # Adjust each stock's position
        res1 = w.bktquery("Capital", "%s 9:40:00" % dt)
        res2 = w.bktquery("Position", "%s 9:40:00" % dt)

        # map(lambda x, y: dict(zip(x, y)), [h]*9, zip(*[iter(v)]*7))
        headers = res2.Fields
        flatten_values = res2.Data
        num_hdrs = len(headers)
```

VN.PY

- 丰富的Python交易和数据API接口
- 事件驱动引擎（vn.event）
- 开发示例（vn.demo）
- 交易平台（vn.trader）
- RPC模块（vn.rpc）
- 官方网站（www.vnpy.org）/知乎专栏
- 官方交流QQ群

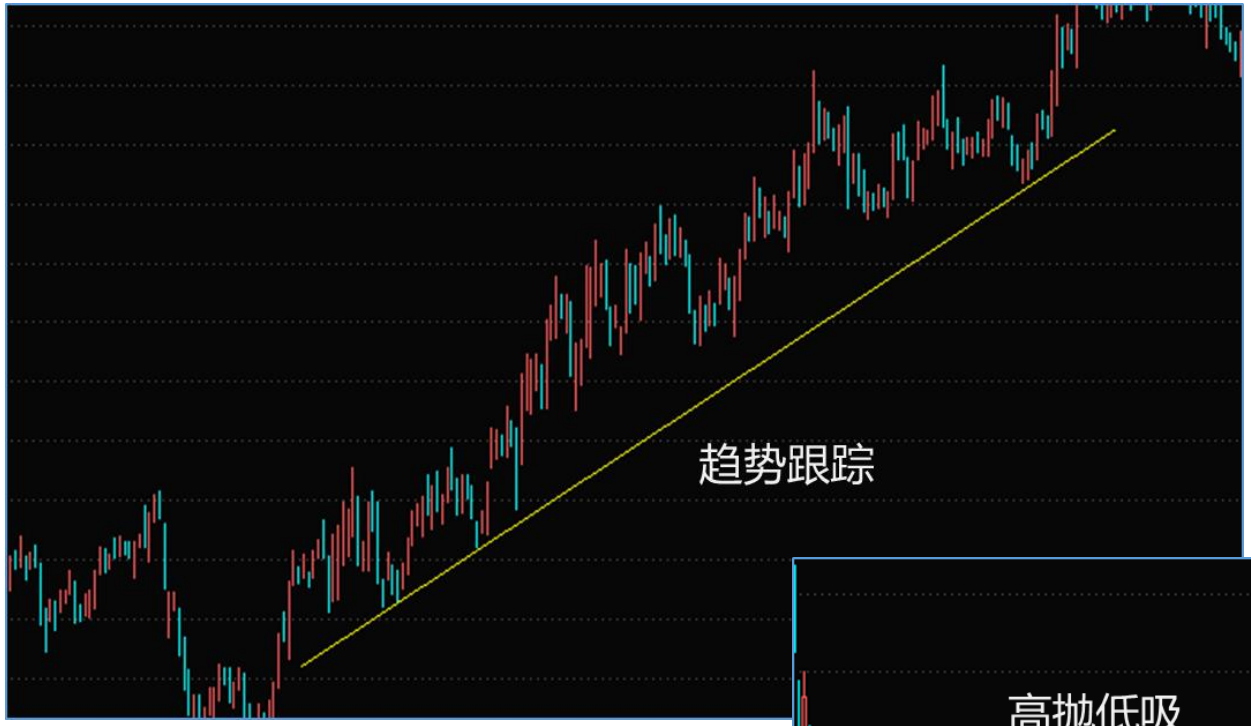

eventEngine.py: vt.wpr: Wing IDE
File Edit Source Refactor Project Debug Testing Git Tools Window Help
eventEngine.py vtEngine.py
MainEngine __init__
170
171 #--
172 def exit(self):
173 """退出程序前调用，保证正常退出"""
174 # 安全关闭所有接口
175 for gateway in self.gatewayDict.values():
176 gateway.close()
177
178 # 停止事件引擎
179 self.eventEngine.stop()
180
181 # 保存数据引擎里的合约数据到硬盘
182 self.dataEngine.saveContracts()
183
184 #--
185 def writeLog(self, content):
186 """快速发出日志事件"""
187 log = VtLogData()
188 log.logContent = content
189 event = Event(type_=EVENT_LOG)
190 event.dict_['data'] = log
191 self.eventEngine.put(event)
192
193 #--
194 def dbConnect(self):
195 """连接MongoDB数据库"""
196 if not self.dbClient:
197 try:
198 self.dbClient = MongoClient()
199 self.writeLog(u'MongoDB连接成功')
200 except ConnectionFailure:
201 self.writeLog(u'MongoDB连接失败')
202
203 #--
eventEngine.py vtEngine.py
test simpletest
12 from eventType import *
13
14
15 ##
16 class EventEngine(object):
17 """
18 事件驱动引擎
19
20 事件驱动引擎中所有的变量都设置为了私有，这是为了防止不小心
21 从外部修改了这些变量的值或状态，导致bug。
22
23 变量说明
24 __queue：私有变量，事件队列
25 __active：私有变量，事件引擎开关
26 __thread：私有变量，事件处理线程
27 __timer：私有变量，计时器
28 __handlers：私有变量，事件处理函数字典
29
30
31 方法说明
32 __run：私有方法，事件处理线程连续运行用
33 __process：私有方法，处理事件，调用注册在引擎中的监听函数
34 __onTimer：私有方法，计时器固定事件间隔触发后，向事件队列中存入计时器事件
35 start：公共方法，启动引擎
36 stop：公共方法，停止引擎
37 register：公共方法，向引擎中注册监听函数
38 unregister：公共方法，向引擎中注销监听函数
39 put：公共方法，向事件队列中存入新的事件
40
41 事件监听函数必须定义为输入参数仅为一个event对象，即：
42
43 函数
44 def func(event)
45 ...
Project Source Browser Snippets Source Assistant Call Stack Indentation
Search in Files Debug I/O Search Stack Data Exceptions Breakpoints Testing Uses
Debug Probe Watch Modules Bookmarks Messages OS Commands
Line 322 Col 10 - [User]

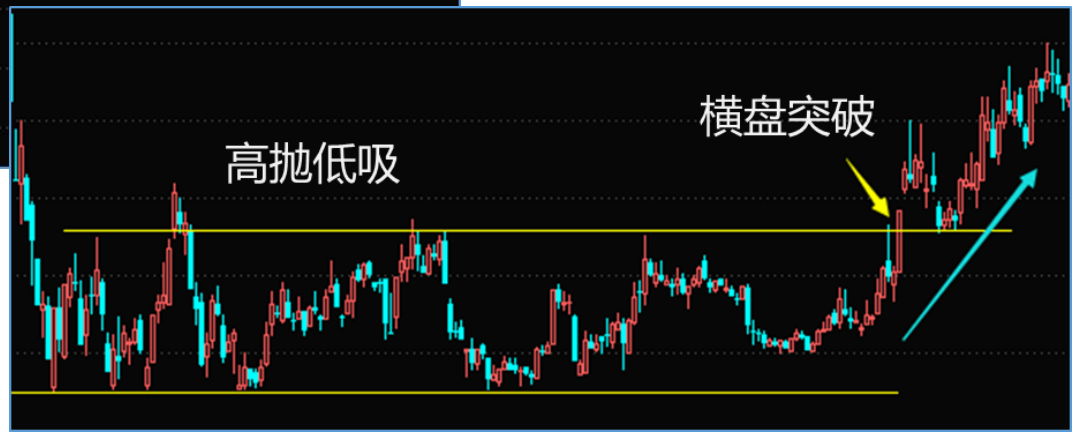

CTA策略
加载策略
全部初始化
全部启动
全部停止
double ema
初始化
启动
停止
name
className
author
vtSymbol
fastK
double ema
DoubleEm...
用Python...
IF1602
0.9
inited
trading
pos
fastMa0
fastMa1
False
False
0
0.0
0.0
21:43:35 CTA引擎启动成功
21:43:36 double ema的交易合约IF1602无法找到
21:43:36 策略加载成功
VnTrader
系统 功能 算法 帮助
代码 p1609
名称 棕榈油p1609
方向类型 多
开平 开仓
价格 0.0000
数量 0
价格类型 限价
交易所
货币
产品类型
交易接口
发单
全撤
卖五
卖四
卖三
卖二
卖一 5518.0 1355
最新 5516.0 -0.04%
买一 5516.0 263
买二
买三
买四
买五
合约代码 名称 最新价 成交量 持仓量 开盘价 最高价 最低价 买一价 买一量 卖一价 卖一量
p1609 棕榈油p1609 5516.0 200268 788240 5504.0 5520.0 5482.0 5516.0 263 5518.0 1355
p1608 棕榈油p1608 5476.0 6 5464.0 1 5570.0 1
p1607 棕榈油p1607 5510.0 2 5400.0 1 5498.0 1
p1606 棕榈油p1606 5436.0 14 5338.0 1 5434.0 1
p1605 棕榈油p1605 5426.0 9212 145462 5430.0 5432.0 5406.0 5426.0 311 5428.0 84
m1609 豆粕m1609 2341.0 177590 2653822 2340.0 2346.0 2337.0 2341.0 1349 2342.0 1187
m1608 豆粕m1608 2351.0 146 2348.0 6 2354.0 1
m1607 豆粕m1607 2368.0 324 2345.0 3 2362.0 6
m1605 豆粕m1605 2306.0 4742 421072 2305.0 2309.0 2302.0 2305.0 94 2306.0 12
IF1609 沪深300IF1609 3015.0 458 4051 3015.4 3023.2 2994.6 3013.2 1 3017.4 1
IF1606 沪深300IF1606 3139.0 1100 7759 3139.6 3144.8 3124.0 3139.0 1 3142.0 2
IF1605 沪深300IF1605 3193.8 1040 1602 3191.0 3199.0 3179.6 3191.2 1 3194.6 1
IF1604 沪深300IF1604 3239.4 14521 35953 3227.8 3244.4 3222.2 3239.0 2 3239.2 1
日志 错误 账户
账户 昨结 净值 可用 手续费 保证金 平仓盈亏 持仓盈亏 接口
000200002874 9619242413.52 9613794263.52 9528094488.52 85699771.0 -5448150.0 XSPEED
成交 委托 持仓
合约代码 名称 方向
p1605 棕榈油p1605 空 1700
i1606 铁矿石i1606 空 30
i1606 铁矿石i1606 多 1
i1605 铁矿石i1605 空 37010
i1605 铁矿石i1605 多 9

量化交易策略的“完整性”

交易系统的核心要素

常见的交易策略类型





交易系统的核心要素

- 市场：买卖什么？
- 头寸规模：买卖多少？
- 入市：什么时候买进？
- 止损：什么时候放弃一个亏损的头寸？
- 退出：什么时候退出一个盈利的头寸？
- 战术：怎么买卖？

以“海龟交易法则”为例

趋势型策略原理


海龟交易系统就是一个
包含了上述所有要素的
”完整的“交易系统

头寸规模计算

- 波动性N：N =（19*PDN+TR）/20
  - PDN = 前一日的N值
  - TR = 当日的真实波动幅度
- 真实波幅 = Max（H-L, |H-PDC|, |PDC-L|）
  - H = 当日最高价
  - L = 当日最低价
  - PDC = 前一日收盘价
- 头寸规模单位（手数）
  = 账户的 1% /（N * 每一最小交易单位）

（举例）

- 假设股票中国平安的N值（≈ATR）为5元
  - 账户的本金为100万元，那么账户的1%就是10000元
  - 最小交易单位是1手股票，也就是100股
- 那么对于中国平安这只股票，可以分配的头寸规模单位是：
  - 10000/（5*100）= 20（手股票）
- 即：给中国平安分配的头寸规模单位是20手

入市策略　　逐步建仓

- 系统1：以20日突破为基础的短期系统
  - 如上次突破是赢利性突破，那么当前的入市信号将被忽略
  - 55日突破点保障信号
- 系统2：以55日突破为基础的长期系统

- 在突破点建立1个单位的头寸
  - 按N/2的价格间隔逐步扩大头寸
  - 以上一份订单的实际成交价为基准
- 直到总头寸达到规模上限

止损

- 任何一笔交易风险程度不超过账户的2%
- 价格变动上限是2N
- 对于加仓情况，止损点上浮

退出

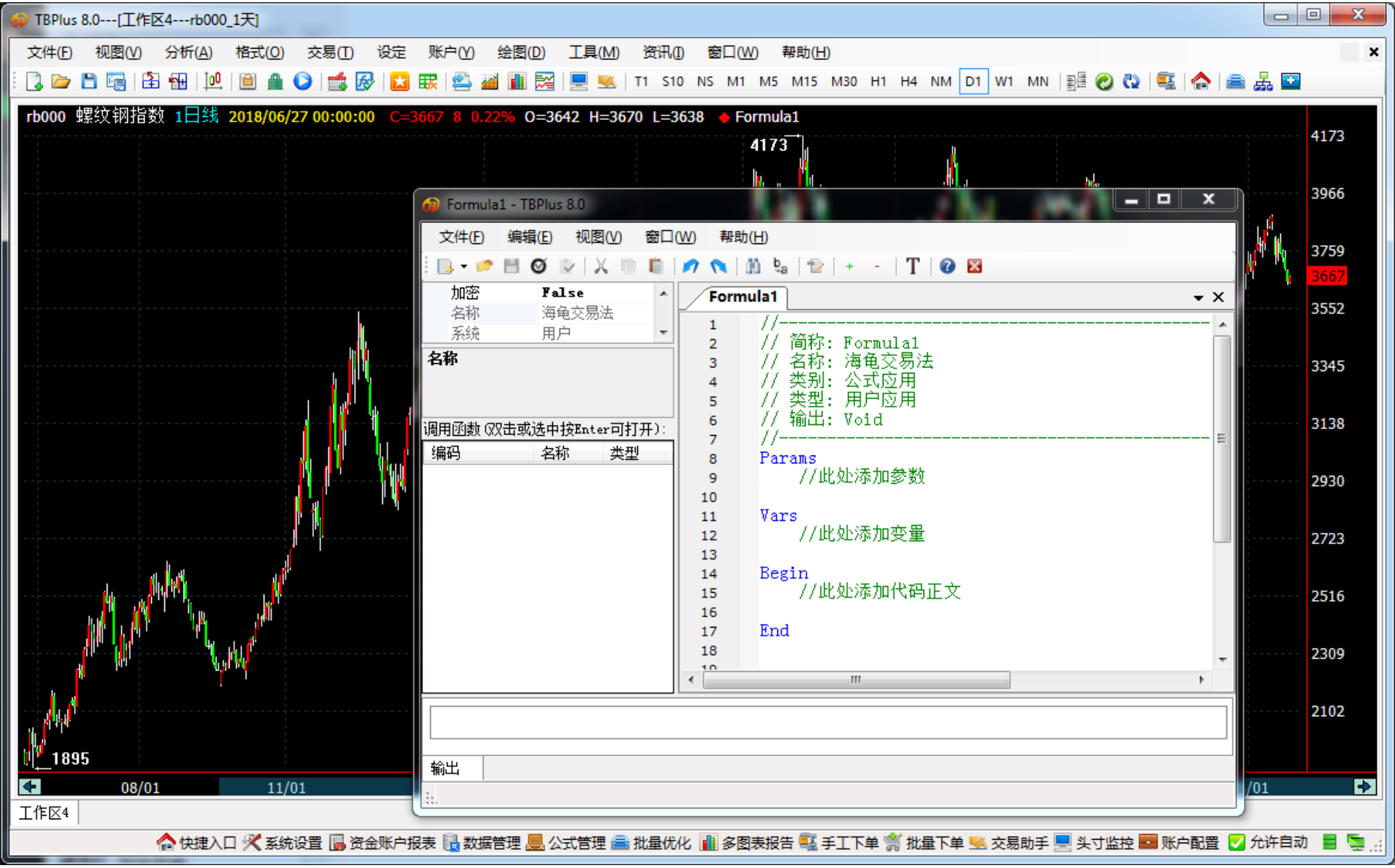
- 系统1：
  - 10日反向突破退出
- 系统2：
  - 20日反向突破退出

动手实践，从策略思想到编程实现，逐步求精

用TB开发商品期货趋势型策略

环境准备

- 运行TB，登录，新建工作区
- 加载交易品种：螺纹钢（指数），日K级别
- 设置商品，增加样本（K线）数到3000
- 设置公式应用，允许连续建仓，最多10次，最大持仓限制增加到2000合约
- 新建“用户应用”Formula1；文件菜单，公式管理，找到Formula1，选中并执行编译
- 插入公式应用，加载Formula1
- 如下页图，开始代码编辑……

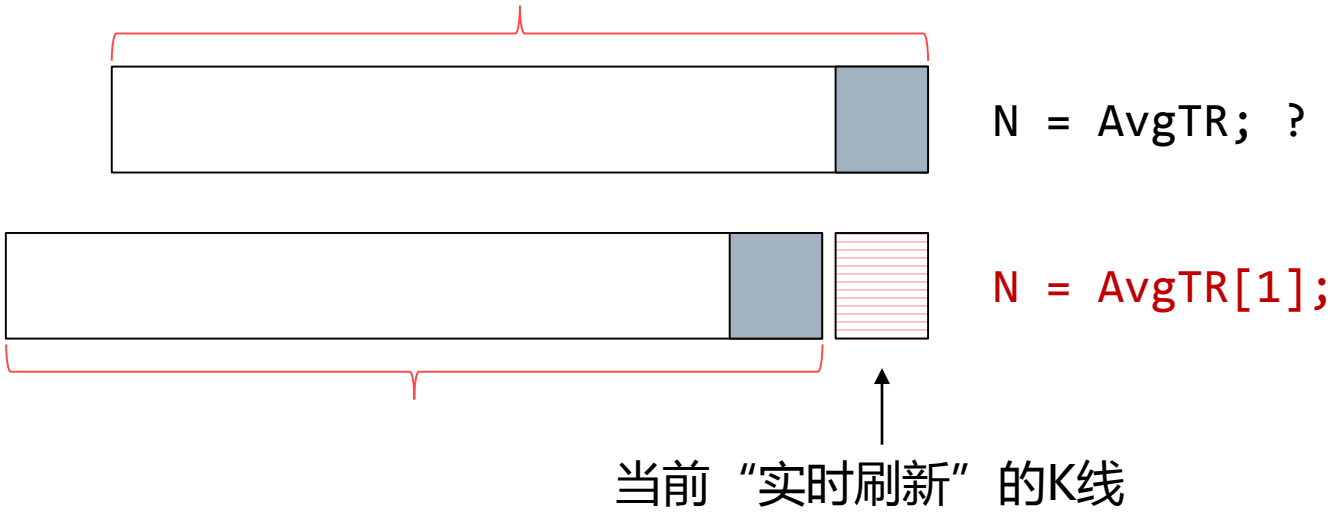

TBPlus 8.0---[工作区4---rb000_1天]
文件(F) 视图(V) 分析(A) 格式(O) 交易(T) 设定 账户(Y) 绘图(D) 工具(M) 资讯(I) 窗口(W) 帮助(H)
T1 S10 NS M1 M5 M15 M30 H1 H4 NM D1 W1 MN
rb000 螺纹钢指数 1日线 2018/06/27 00:00:00 C=3667 8 0.22% O=3642 H=3670 L=3638 Formula1
4173
3966
3759
3667
3552
3345
3138
2930
2723
2516
2309
2102
1895
08/01
11/01
Formula1 - TBPlus 8.0
文件(F) 编辑(E) 视图(V) 窗口(W) 帮助(H)
加密 False
名称 海龟交易法
系统 用户
名称
调用函数(双击或选中按Enter可打开):
编码 名称 类型
Formula1
1 //---
2 // 简称: Formula1
3 // 名称: 海龟交易法
4 // 类别: 公式应用
5 // 类型: 用户应用
6 // 输出: Void
7 //---
8 Params
9 //此处添加参数
10
11 Vars
12 //此处添加变量
13
14 Begin
15 //此处添加代码正文
16
17 End
18
输出
工作区4
快捷入口 系统设置 资金账户报表 数据管理 公式管理 批量优化 多图表报告 手工下单 批量下单 交易助手 头寸监控 账户配置 允许自动

初步实现

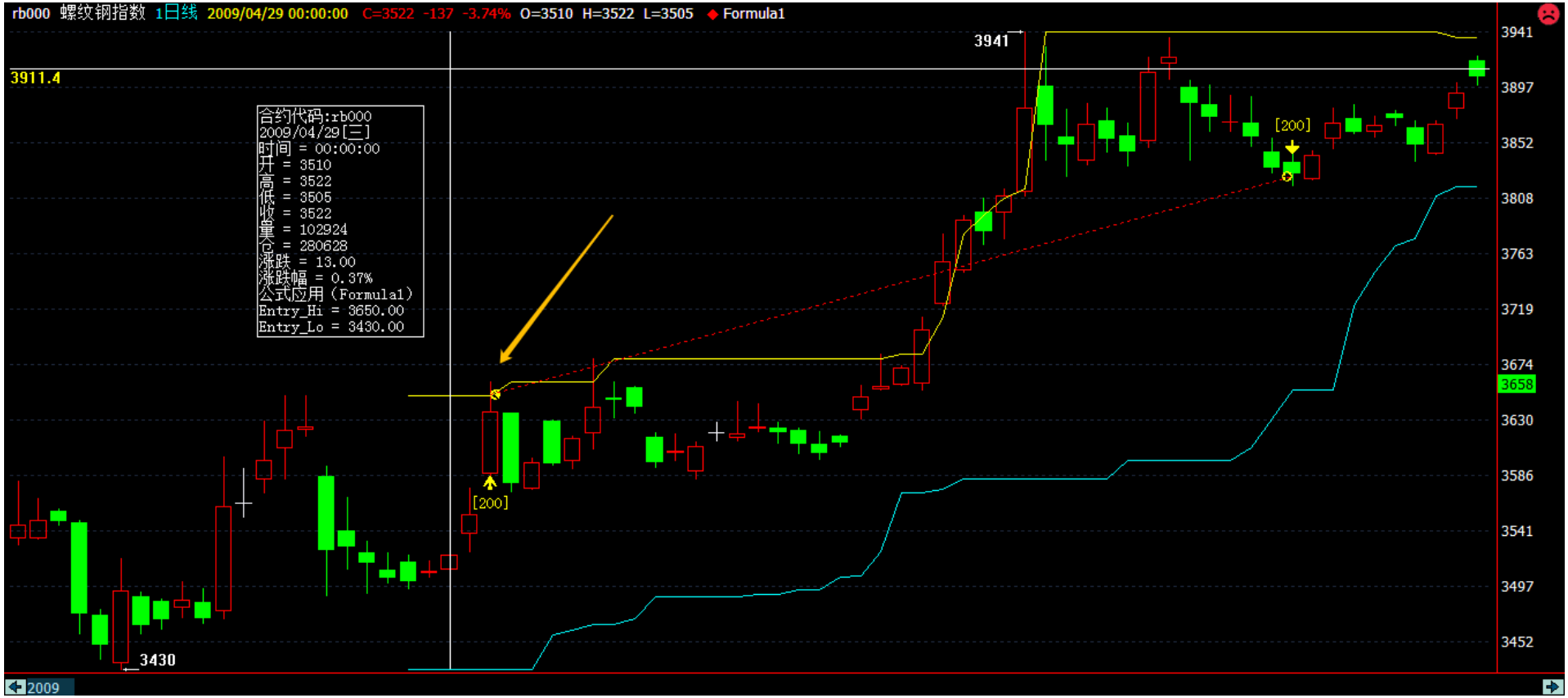
- 建仓信号
  - 突破（20日）前高：开多信号
  - 跌破（20日）前低：开空信号
- 仓位计算（无持仓情况下）
  - 突破时建立一个初始的头寸单位
- 退出（有持仓情况下）
  - 跌破（10日）前低时平多
  - 突破（10日）前高时平空

序列变量平移取值

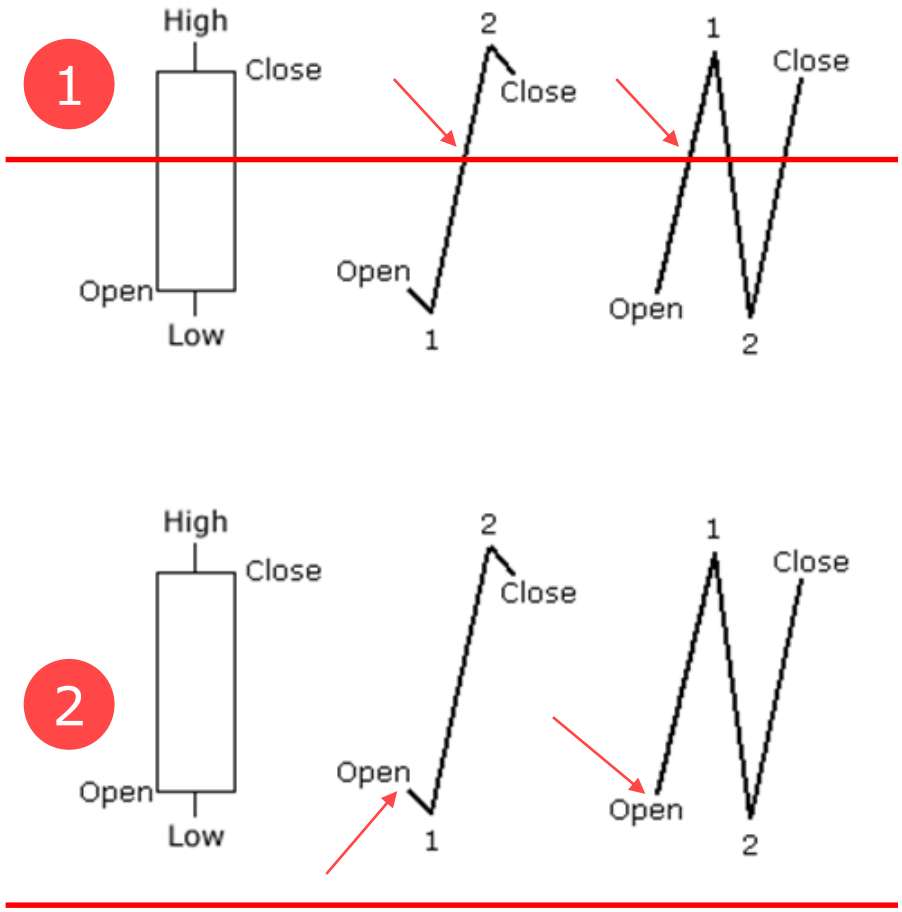
```
AvgTR = XAverage(TrueRange, ATRLength);
N = AvgTR[1];
```



突破入场示意图


rb000 螺纹钢指数 1日线 2009/04/29 00:00:00 C=3522 -137 -3.74% O=3510 H=3522 L=3505 Formula1
3911.4
合约代码:rb000
2009/04/29[三]
时间 = 00:00:00
开 = 3510
高 = 3522
低 = 3505
收 = 3522
量 = 102924
仓 = 280628
涨跌 = 13.00
涨跌幅 = 0.37%
公式应用(Formula1)
Entry_Hi = 3650.00
Entry_Lo = 3430.00
3941
3430
[200]
[200]
3941
3897
3852
3808
3763
3719
3674
3658
3630
3586
3541
3497
3452
2009

入场点代码解析



DonchianHi

```
// 多头开仓
if (high > DonchianHi && TurtleUnits >= 1)
{
    myEntryPrice = min(high,
                       DonchianHi+MinPoint);

    myEntryPrice = IIF(myEntryPrice < Open,
                       Open,
                       myEntryPrice); // 应对跳空

    Buy(TurtleUnits, myEntryPrice);
}
```

DonchianHi

休息一下

5分钟后回来

```
Params
    Numeric RiskRatio(1);  // 1% risk ratio
    Numeric ATRLength(20); // ATR window size
    Numeric boLength(20);  // breakout
    Numeric teLength(10);  // trailing exit
Vars
    Numeric N;
    NumericSeries AvgTR;
    Numeric TotalEquity;
    Numeric TurtleUnits;
    NumericSeries DonchianHi;
    NumericSeries DonchianLo;
    Numeric MinPoint;
    Numeric ExitLowestPrice;
    Numeric ExitHighestPrice;
    Numeric myEntryPrice;
    Numeric myExitPrice;
Begin
    if (!CallAuctionFilter()) return;

    MinPoint = MinMove * PriceScale;
    AvgTR = XAverage(TrueRange, ATRLength);
    N = AvgTR[1];
    TotalEquity = Portfolio_CurrentCapital() + Portfolio_UsedMargin();
    TurtleUnits = (TotalEquity*RiskRatio/100) / (N*ContractUnit()*BigPointValue());
    TurtleUnits = IntPart(TurtleUnits);
```

【初步实现的（简化版）代码】

```
DonchianHi = HighestFC(High[1], boLength);
DonchianLo = LowestFC(Low[1], boLength);
// PlotNumeric("Entry_Hi", DonchianHi);
// PlotNumeric("Entry_Lo", DonchianLo);

ExitLowestPrice = LowestFC(Low[1], teLength);
ExitHighestPrice = HighestFC(High[1], teLength);

if (MarketPosition == 0)
{
    // 多头开仓
    if (high > DonchianHi && TurtleUnits >= 1)
    {
        myEntryPrice = min(high, DonchianHi+MinPoint);
        myEntryPrice = IIF(myEntryPrice < Open, Open, myEntryPrice);  // 应对跳空情况
        Buy(TurtleUnits, myEntryPrice);
    }
    // 空头开仓
    if (Low < DonchianLo && TurtleUnits >= 1)
    {
        myEntryPrice = max(low, DonchianLo-MinPoint);
        myEntryPrice = IIF(myEntryPrice > Open, Open, myEntryPrice);  // 应对跳空情况
        SellShort(TurtleUnits, myEntryPrice);
    }
}
```

```
    if (MarketPosition == 1)
    {
        if (Low < ExitLowestPrice)
        {
            // 多头退出
            myExitPrice = max(Low, ExitLowestPrice-MinPoint);
            myExitPrice = IIF(myExitPrice > Open, Open, myExitPrice); // 应对跳空情况
            Sell(0, myExitPrice);
        }
        else {
            // 多头加仓逻辑... 多头止损逻辑...
        }
    }
    else if (MarketPosition == -1)
    {
        if (High > ExitHighestPrice)
        {
            // 空头退出
            myExitPrice = min(High, ExitHighestPrice+MinPoint);
            myExitPrice = IIF(myExitPrice < Open, Open, myExitPrice); // 应对跳空情况
            BuyToCover(0, myExitPrice);
        }
        else {
            // 空头加仓逻辑... 空头止损逻辑...
        }
    }
End
```

【简化版回测结果】

【螺纹钢】总收益：263.10%，最大回撤：14.98%，胜率：51.76%，盈亏比：2.49

完整逻辑

- 增仓动作（有持仓情况下）
  - 多头：相对于上次入场价格，每上涨N/2就新增一个单位的多头头寸，每根K线内可多次加仓
  - 空头：相对于上次入场价格，每下跌N/2就新增一个单位的空头头寸，每根K线内可多次加仓
- 退出（有持仓情况下）
  - 新增规则：相对于上次入场价格，如超过最大允许的损失幅度（2*N），则清仓
  - 加仓所在的那根K线内，不进行止损

其它细节

- 长周期突破点保障信号
  - 入市过滤条件
- 前次突破失败与否的判别
  - 盈利性退出
  - 亏损性退出

```
if (MarketPosition == 0 && \
   (!LastProfitableTradeFilter or PreBreakoutFailure))
{
    // ...
}
```

```
Params
    Numeric RiskRatio(1);                    // % Risk Per N ( 0 - 100)
    Numeric ATRLength(20);                   // 平均波动周期 ATR Length
    Numeric boLength(20);                    // 短周期 BreakOut Length
    Numeric fsLength(55);                    // 长周期 FailSafe Length
    Numeric teLength(10);                    // 离市周期 Trailing Exit Length
    Bool LastProfitableTradeFilter(True);    // 使用入市过滤条件
Vars
    Numeric MinPoint;                        // 最小变动单位
    NumericSeries AvgTR;                     // ATR
    Numeric N;                               // N 值
    Numeric TotalEquity;                     // 按最新收盘价计算出的总资产
    Numeric TurtleUnits;                     // 交易单位
    NumericSeries DonchianHi;                // 唐奇安通道上轨，延后1个Bar
    NumericSeries DonchianLo;                // 唐奇安通道下轨，延后1个Bar
    NumericSeries fsDonchianHi;              // 唐奇安通道上轨，延后1个Bar，长周期
    NumericSeries fsDonchianLo;              // 唐奇安通道下轨，延后1个Bar，长周期
    Numeric ExitHighestPrice;                // 离市时判断需要的N周期最高价
    Numeric ExitLowestPrice;                 // 离市时判断需要的N周期最低价
    Numeric myEntryPrice;                    // 开仓价格
    Numeric myExitPrice;                     // 平仓价格
    Bool SendOrderThisBar(False);            // 当前Bar有过交易
    NumericSeries preEntryPrice(0);          // 前一次开仓的价格
    BoolSeries PreBreakoutFailure(false);    // 前一次突破是否失败
Begin
    // 集合竞价过滤
    if (!CallAuctionFilter()) return;
```

【TB自带示例（完整版）代码】

```
If(BarStatus == 0)
{
    preEntryPrice = InvalidNumeric;
    PreBreakoutFailure = false;
}

MinPoint = MinMove*PriceScale;
AvgTR = XAverage(TrueRange,ATRLength);
N = AvgTR[1];
TotalEquity = Portfolio_CurrentCapital() + Portfolio_UsedMargin();
TurtleUnits = (TotalEquity*RiskRatio/100) /(N * ContractUnit()*BigPointValue());
TurtleUnits = IntPart(TurtleUnits); // 对小数取整

DonchianHi = HighestFC(High[1],boLength);
DonchianLo = LowestFC(Low[1],boLength);

fsDonchianHi = HighestFC(High[1],fsLength);
fsDonchianLo = LowestFC(Low[1],fsLength);

ExitLowestPrice = LowestFC(Low[1],teLength);
ExitHighestPrice = HighestFC(High[1],teLength);

Commentary("N="+Text(N));
Commentary("preEntryPrice="+Text(preEntryPrice));
Commentary("PreBreakoutFailure="+IIFString(PreBreakoutFailure,"True","False"));
```

```
// 当不使用过滤条件，或者使用过滤条件并且条件为PreBreakoutFailure为True进行后续操作
If(MarketPosition == 0 && ((!LastProfitableTradeFilter) Or (PreBreakoutFailure)))
{
    // 突破开仓
    If(High > DonchianHi && TurtleUnits >= 1)
    {
        // 开仓价格取突破上轨+一个价位和最高价之间的较小值，更接近真实情况，并能尽量保证成交
        myEntryPrice = min(high,DonchianHi + MinPoint);
        myEntryPrice = IIF(myEntryPrice < Open, Open,myEntryPrice); // 跳空用开盘价代替
        preEntryPrice = myEntryPrice;
        Buy(TurtleUnits,myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }

    If(Low < DonchianLo && TurtleUnits >= 1)
    {
        // 开仓价格取突破下轨-一个价位和最低价之间的较大值，更接近真实情况，并能尽量保证成交
        myEntryPrice = max(low,DonchianLo - MinPoint);
        myEntryPrice = IIF(myEntryPrice > Open, Open,myEntryPrice); // 跳空用开盘价
        preEntryPrice = myEntryPrice;
        SendOrderThisBar = True;
        SellShort(TurtleUnits,myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }
}
```

```
// 长周期突破开仓 Failsafe Breakout point
If(MarketPosition == 0)
{
    Commentary("fsDonchianHi="+Text(fsDonchianHi));
    If(High > fsDonchianHi && TurtleUnits >= 1)
    {
        // 开仓价格取突破上轨+一个价位和最高价之间的较小值，更接近真实情况，并能尽量保证成交
        myEntryPrice = min(high,fsDonchianHi + MinPoint);
        myEntryPrice = IIF(myEntryPrice < Open, Open,myEntryPrice); // 跳空用开盘价代替
        preEntryPrice = myEntryPrice;
        Buy(TurtleUnits,myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }

    Commentary("fsDonchianLo="+Text(fsDonchianLo));
    If(Low < fsDonchianLo && TurtleUnits >= 1)
    {
        // 开仓价格取突破下轨-一个价位和最低价之间的较大值，更接近真实情况，并能尽量保证成交
        myEntryPrice = max(low,fsDonchianLo - MinPoint);
        myEntryPrice = IIF(myEntryPrice > Open, Open,myEntryPrice); // 跳空用开盘价代替
        preEntryPrice = myEntryPrice;
        SellShort(TurtleUnits,myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }
}
```

```
If(MarketPosition == 1) // 有多仓的情况
{
    Commentary("ExitLowestPrice="+Text(ExitLowestPrice));
    If(Low < ExitLowestPrice)
    {
        myExitPrice = max(Low,ExitLowestPrice - MinPoint);
        myExitPrice = IIF(myExitPrice > Open, Open,myExitPrice); // 跳空用开盘价代替
        Sell(0,myExitPrice);    // 数量用0的情况下将全部平仓
    }Else
    {
        If(preEntryPrice!=InvalidNumeric && TurtleUnits >= 1)
        {
            If(Open >= preEntryPrice + 0.5*N) // 如开盘超过设定的1/2N,则直接用开盘价增仓
            {
                myEntryPrice = Open;
                preEntryPrice = myEntryPrice;
                Buy(TurtleUnits,myEntryPrice);
                SendOrderThisBar = True;
            }

            while(High >= preEntryPrice + 0.5*N) // 以最高价为标准，判断能进行几次增仓
            {
                myEntryPrice = preEntryPrice + 0.5 * N;
                preEntryPrice = myEntryPrice;
                Buy(TurtleUnits,myEntryPrice);
                SendOrderThisBar = True;
            }
        }
```

```
        // 止损指令
        If(Low <= preEntryPrice - 2 * N && SendOrderThisBar == false) // 加仓Bar不止损
        {
            myExitPrice = preEntryPrice - 2 * N;
            myExitPrice = IIF(myExitPrice > Open, Open,myExitPrice); // 跳空用开盘价代替
            Sell(0,myExitPrice); // 数量用0的情况下将全部平仓
            PreBreakoutFailure = True;
        }
    }
}Else If(MarketPosition ==-1) // 有空仓的情况
{
    // 求出持空仓时离市的条件比较值
    Commentary("ExitHighestPrice="+Text(ExitHighestPrice));
    If(High > ExitHighestPrice)
    {
        myExitPrice = Min(High,ExitHighestPrice + MinPoint);
        myExitPrice = IIF(myExitPrice < Open, Open,myExitPrice); // 跳空用开盘价代替
        BuyToCover(0,myExitPrice);    // 数量用0的情况下将全部平仓
    }Else
    {
        If(preEntryPrice!=InvalidNumeric && TurtleUnits >= 1)
        {
            If(Open <= preEntryPrice - 0.5*N) // 如开盘超过设定的1/2N,则直接用开盘价增仓
            {
```

```
                myEntryPrice = Open;
                preEntryPrice = myEntryPrice;
                SellShort(TurtleUnits,myEntryPrice);
                SendOrderThisBar = True;
            }

            while(Low <= preEntryPrice - 0.5*N) // 以最低价为标准，判断能进行几次增仓
            {
                myEntryPrice = preEntryPrice - 0.5 * N;
                preEntryPrice = myEntryPrice;
                SellShort(TurtleUnits,myEntryPrice);
                SendOrderThisBar = True;
            }
        }

        // 止损指令
        If(High >= preEntryPrice + 2 * N &&SendOrderThisBar==false) // 加仓Bar不止损
        {
            myExitPrice = preEntryPrice + 2 * N;
            myExitPrice = IIF(myExitPrice < Open, Open,myExitPrice); // 跳空用开盘价代替
            BuyToCover(0,myExitPrice); // 数量用0的情况下将全部平仓
            PreBreakoutFailure = True;
        }
    }
}
End
```


4870
[2000]
[457]
[756]
[787]
[2000]
[282]
[564]
[594]
[560]
[836]
[836]
[378]
[2000]
1895
4972
[2000]
[234]
[281]
[301]
[281]
[301]
[301]
[301]

【完整版回测结果】

【螺纹钢】总收益：700.22%，最大回撤：21.79%，胜率：50.09%，盈亏比：2.01

对比 -【简化版回测结果】

【螺纹钢】总收益：263.10%，最大回撤：14.98%，胜率：51.76%，盈亏比：2.49

思考

- 完整实现的效果一定比简化版好吗？
- 思考问题的本质
  - 突破：趋势型策略，顺势加仓，在什么情况下回调风险大？
  - 对品种、周期的选择有什么要求？
- 尝试改写完整版的代码？
  - 同时进行跨市场、多品种的交易
- 测试不同级别：60分钟、30分钟、周线
- 测试多个品种：RB -> J，RU，PTA

期货交易策略的注意事项

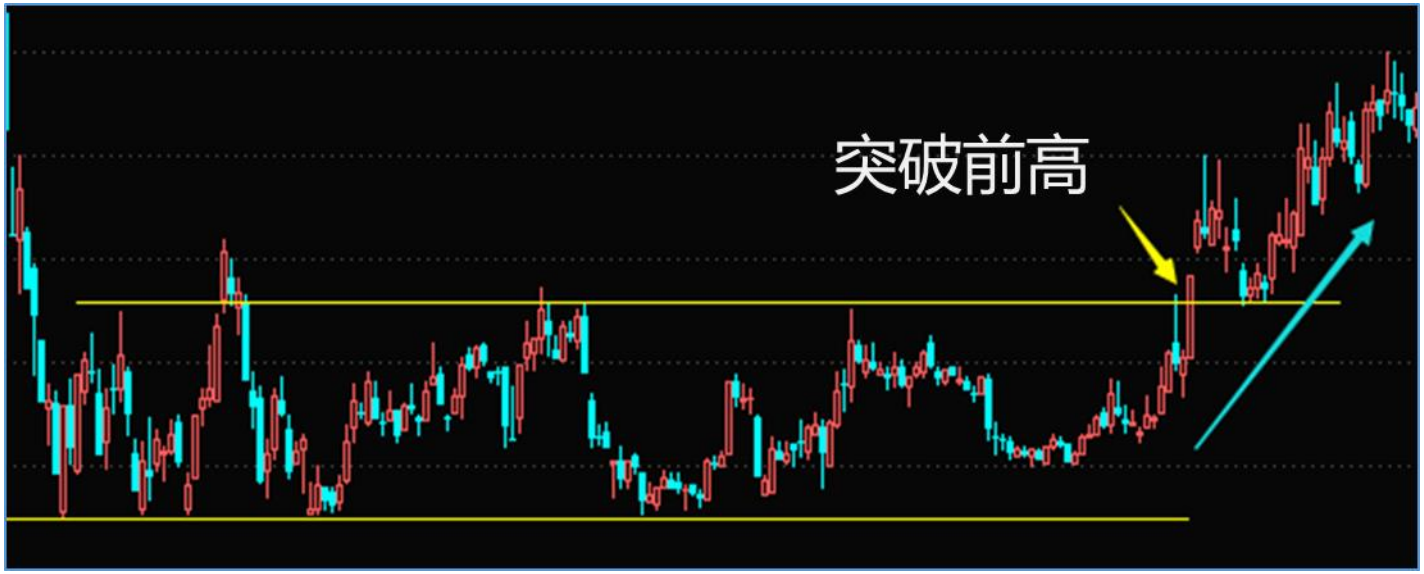
- 移仓换月（主力合约切换）
- 杠杆效应（保证金交易）
- 波动幅度（隔夜跳空风险）
- 套利机会（跨期、跨品种）

用Python + Tushare免费数据源

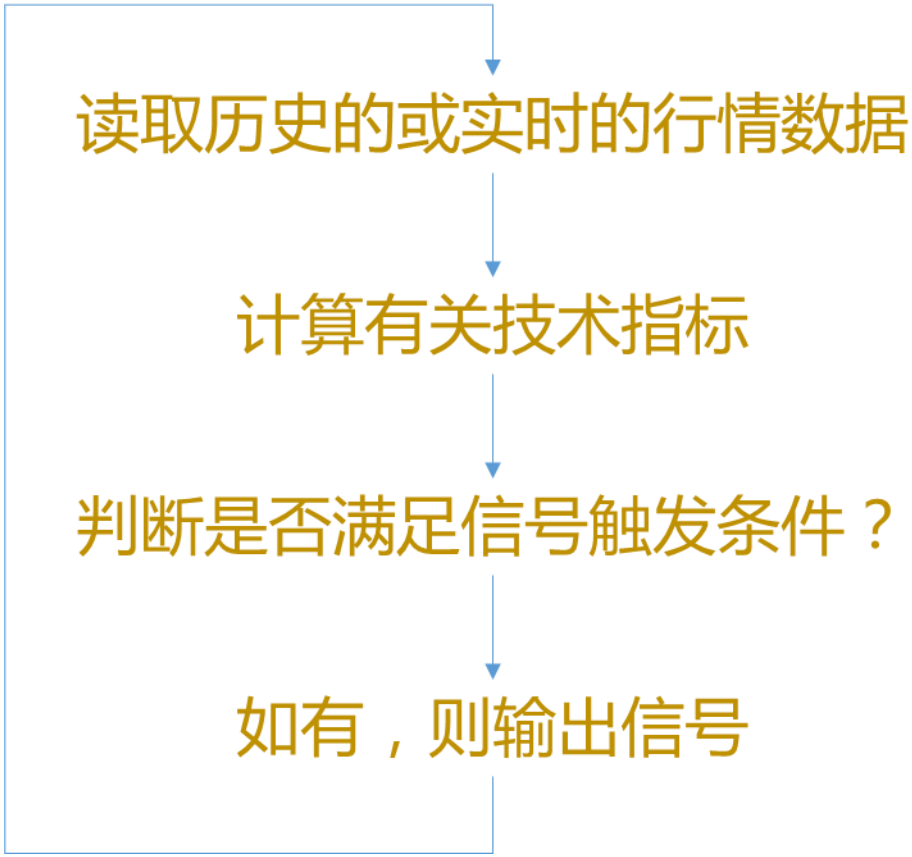
编写一个股票突破信号系统

示例：突破信号

- 突破55日前高，作为买入信号
- 细节：当日突破/前日未突破；日线复权处理

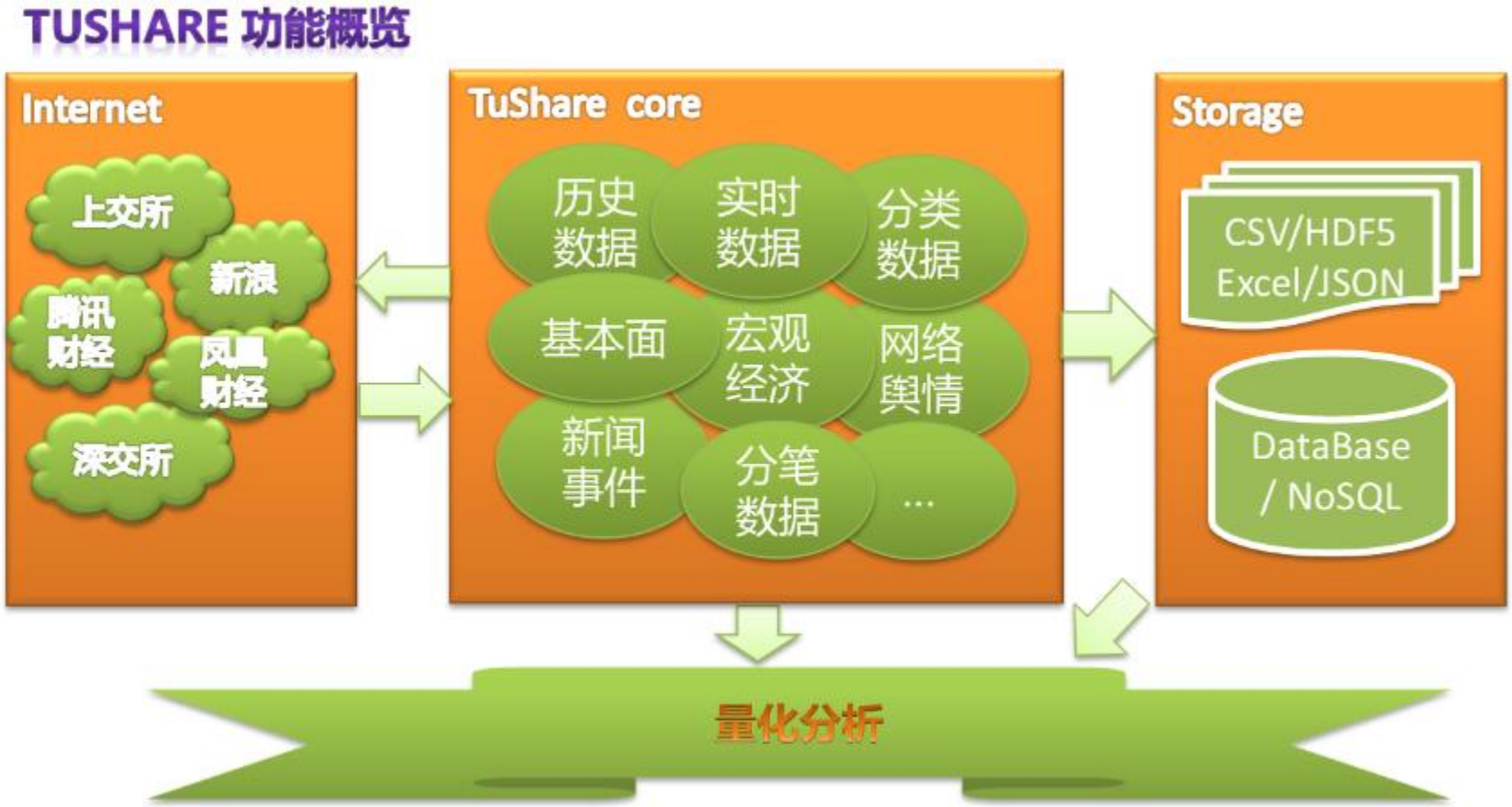


检测信号的一般代码流程



对每只股票分别执行该操作

行情数据源： http://tushare.org



单只股票的信号检测

```
# -*- coding: utf-8 -*-

import tushare as ts

window_size = 5  # 过程演示用，待改为55
code = "000001"

temp = ts.get_k_data(code, start="2018-03-01", ktype="D", autype="qfq")
temp.index = temp.pop("date")
df = temp.loc[:, ["high", "close"]]
df["hhv"] = df["high"].rolling(window_size).max()
df["pre_hhv"] = df["hhv"].shift(1)
# df["signals"] = df["close"] > df["pre_hhv"]
df["signals"] = (df["close"].shift(1) <= df["pre_hhv"].shift(1)) & \
                (df["close"] > df["pre_hhv"])
print(df)
results = df[df["signals"]]
print(list(results.index))
```

date	high	close	hhv	pre_hhv	signals
2018-03-01	12.15	12.04	NaN	NaN	False
2018-03-02	12.04	11.95	NaN	NaN	False
2018-03-05	12.08	11.86	NaN	NaN	False
2018-03-06	12.11	12.10	NaN	NaN	False
2018-03-07	12.34	12.05	12.34	NaN	False
2018-03-08	12.15	12.11	12.34	12.34	False
2018-03-09	12.20	12.09	12.34	12.34	False
2018-03-12	12.17	12.03	12.34	12.34	False
2018-03-13	12.22	12.02	12.34	12.34	False
2018-03-14	12.00	11.92	12.22	12.34	False
2018-03-15	11.85	11.71	12.22	12.22	False
2018-03-16	11.85	11.64	12.22	12.22	False
2018-03-19	11.84	11.83	12.22	12.22	False
2018-03-20	11.88	11.82	12.00	12.22	False
2018-03-21	12.12	11.90	12.12	12.00	False
2018-03-22	11.97	11.66	12.12	12.12	False
2018-03-23	11.35	11.34	12.12	12.12	False
2018-03-26	11.20	10.93	12.12	12.12	False
2018-03-27	11.17	10.94	12.12	12.12	False
2018-03-28	11.14	10.89	11.97	12.12	False
2018-03-29	11.17	11.05	11.35	11.97	False
2018-03-30	11.05	10.90	11.20	11.35	False
2018-04-02	10.99	10.71	11.17	11.20	False
2018-04-03	10.67	10.56	11.17	11.17	False
2018-04-04	11.01	10.87	11.17	11.17	False
2018-04-09	11.10	11.02	11.10	11.17	False
2018-05-21	11.11	10.95	11.19	11.23	False
2018-05-22	10.98	10.86	11.11	11.19	False
2018-05-23	10.82	10.65	11.11	11.11	False
2018-05-24	10.68	10.61	11.11	11.11	False
2018-05-25	10.67	10.59	11.11	11.11	False
2018-05-28	10.66	10.59	10.98	11.11	False
2018-05-29	10.63	10.38	10.82	10.98	False
2018-05-30	10.29	10.08	10.68	10.82	False
2018-05-31	10.19	10.18	10.67	10.68	False
2018-06-01	10.29	10.19	10.66	10.67	False
2018-06-04	10.31	10.27	10.63	10.66	False
2018-06-05	10.30	10.26	10.31	10.63	False
2018-06-06	10.26	10.14	10.31	10.31	False
2018-06-07	10.46	10.37	10.46	10.31	True
2018-06-08	10.33	10.12	10.46	10.46	False
2018-06-11	10.13	10.04	10.46	10.46	False
2018-06-12	10.08	10.06	10.46	10.46	False
2018-06-13	10.06	9.95	10.46	10.46	False
2018-06-14	10.15	10.07	10.33	10.46	False
2018-06-15	10.29	10.17	10.29	10.33	False
2018-06-19	10.15	9.87	10.29	10.29	False
2018-06-20	9.95	9.91	10.29	10.29	False
2018-06-21	10.04	9.86	10.29	10.29	False
2018-06-22	9.87	9.85	10.29	10.29	False
2018-06-25	9.92	9.46	10.15	10.29	False

[78 rows x 5 columns]
[Finished in 1.5s]

全市场股票的信号检测

```
# -*- coding: utf-8 -*-

import tushare as ts

window_size = 55

all_stocks = ts.get_stock_basics()["name"]
for code, name in all_stocks.iteritems():
    temp = ts.get_k_data(code, start="2018-01-01", ktype="D", autype="qfq")
    temp.index = temp.pop("date")
    df = temp.loc[:, ["high", "close"]]
    df["hhv"] = df["high"].rolling(window_size).max()
    df["pre_hhv"] = df["hhv"].shift(1)
    df["signals"] = (df["close"].shift(1) <= df["pre_hhv"].shift(1)) & \
                    (df["close"] > df["pre_hhv"])
    results = df[df["signals"]]
    if len(results) > 0:
        print(code, name, list(results.index))
```

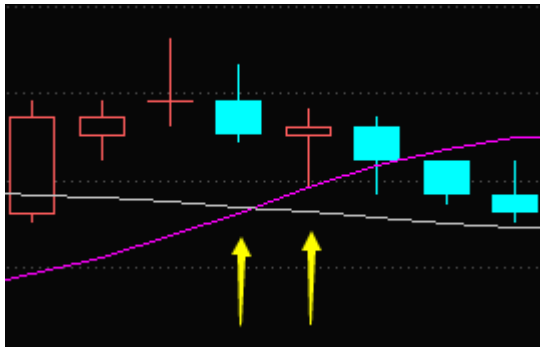
```
300333 兆日科技 ['2018-04-19']
600579 天华院 ['2018-06-05']
300139 晓程科技 ['2018-04-25', '2018-06-19']
300490 华自科技 ['2018-05-29']
002774 快意电梯 ['2018-05-25']
603477 振静股份 ['2018-06-25']
002368 太极股份 ['2018-03-28', '2018-03-30', '2018-04-16', '2018-04-18']
600779 水井坊 ['2018-05-10', '2018-05-16', '2018-06-04']
300689 澄天伟业 ['2018-06-14']
600536 中国软件 ['2018-04-18']
300620 光库科技 ['2018-03-30', '2018-04-11', '2018-05-07']
600756 浪潮软件 ['2018-04-24']
300604 长川科技 ['2018-04-02', '2018-04-18', '2018-04-26']
300384 三联虹普 ['2018-04-12', '2018-05-18']
300170 汉得信息 ['2018-03-29']
002917 金奥博 ['2018-06-15']
002886 沃特股份 ['2018-04-09']
002023 海特高新 ['2018-04-17', '2018-04-26', '2018-05-23']
002896 中大力德 ['2018-06-26']
002796 世嘉科技 ['2018-05-29', '2018-05-31']
300302 同有科技 ['2018-04-02', '2018-04-18']
002134 天津普林 ['2018-04-25']
603977 国泰集团 ['2018-03-29', '2018-04-09']
300062 中能电气 ['2018-05-30']
002802 洪汇新材 ['2018-05-08']
300571 平治信息 ['2018-05-22']
000977 浪潮信息 ['2018-04-16', '2018-04-18']
002741 光华科技 ['2018-03-30', '2018-04-17', '2018-05-10', '2018-05-21', '2018-05-24']
```

总结

- 常用的第三方量化平台
- 完整的交易系统要素
- 从策略原理到编程实现的完整过程
- 平台·语言·数据·信号

课后练习

- 用TB测试海龟交易法在不同周期、不同品种上的表现，撰写测试报告
  - 测试不同级别：60分钟、30分钟、周线
  - 测试多个品种：RB -> J，RU，PTA
  - 对比完整版和简化版的表现，分析原因
- 用Python和Tushare数据源，检测日线级别下的“双均线交叉”信号
  - 上穿（金叉）或下穿（死叉）的动作如何定义？



下节课预告

- 题目：用Python开发均值回复型股票策略

  - 编程语言：Python
  - 运行平台：聚宽 - http://joinquant.com

  - 准备工作：
    - 提前在聚宽上注册账号
    - 阅读并熟悉聚宽的API文档
    - 自学Python和pandas

问答互动

在所报课的课程页面，
1、点击“全部问题”显示本课程所有学员提问的问题。
2、点击“提问”即可向该课程的老师和助教提问问题。

联系我们

小象学院：互联网新技术在线教育领航者

– 微信公众号：**小象学院**

THANKS