★★★

严谨　认真　持续

用Python/金字塔/TB写一个趋势跟随策略

量化投资实战交易体系必修课

系列4 - 快速开发一个量化策略

普量学院
PLOUTO-QUANTS

目录

CONTENT

量化平台的选择

PART ONE

海龟交易法则

PART TWO

三种语言的实例

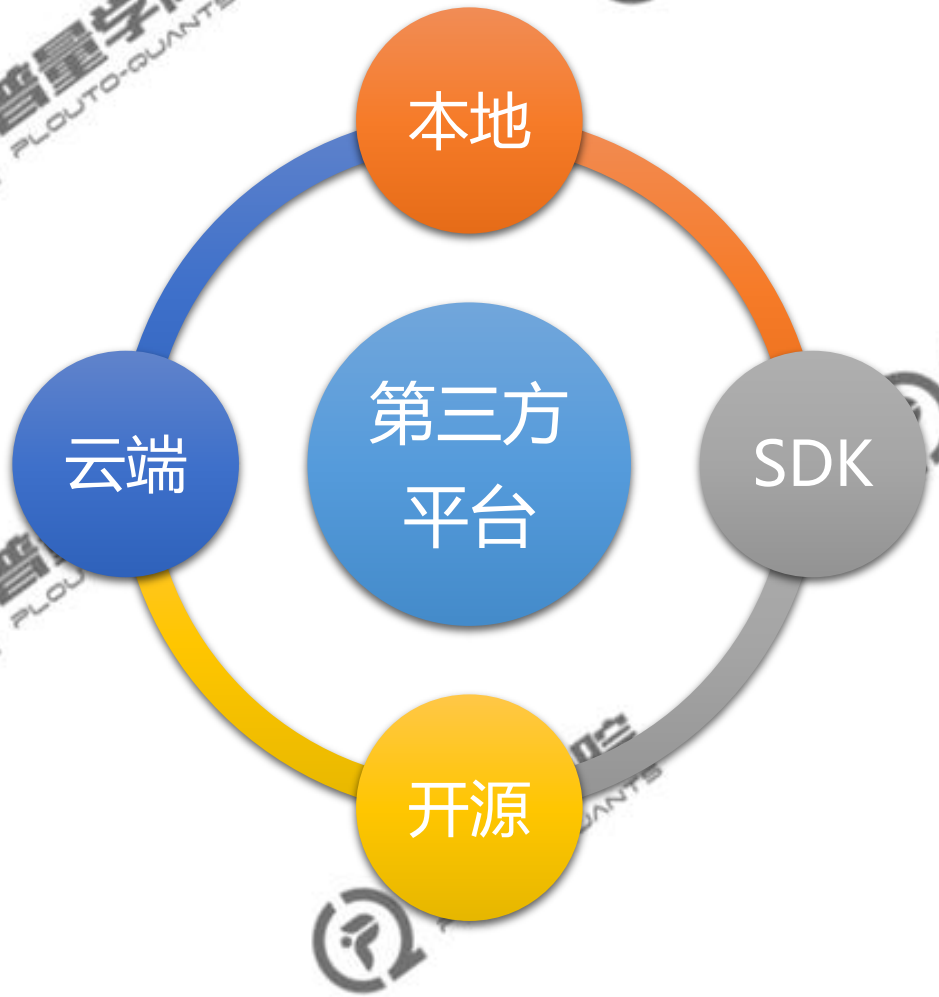
PART THREE

量化平台的选择

PART ONE

一个量化策略的进阶之路

从零开始构建自己的量化系统



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

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)

T1 S10 NS M1 M5 M15 M30 H1 H4 NM D1 W1 MN

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
1	白银指数
2	黄金指数
3	沥青指数
4	沪铝指数
5	螺纹钢指数
6	沪镍指数
7	沪铅指数
8	橡胶指数
9	沪铜指数
10	沪锡指数
11	线材指数
12	沪锌指数
13	热轧卷板指数
14	PVC指数
15	豆一指数
16	豆粕指数
17	豆油指数
18	鸡蛋指数
19	焦煤指数
20	焦炭指数
21	胶合板指数
22	聚丙烯指数
23	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..

2016/08/19[五]
时间 = 21:30:00
价格 = 2565
开 = 2557
高 = 2563
低 = 2554
收 = 2560
量 = 180262
仓 = 3642300
涨幅 = 0.12%
公式应用(DualMA)
MA1 = 2560.00
MA2 = 2547.35

08月19日 11:00 14:30 22:00

rb000 螺纹钢指数 30分钟线 2016/08/19 22:30 C=2565 18 0.71% O=2556 H=2569..

08月12日 08月16日 08月18日

rb000 螺纹钢指数 1小时线 2016/08/19 22:00 C=2565 18 0.71% O=2576 H=2580..

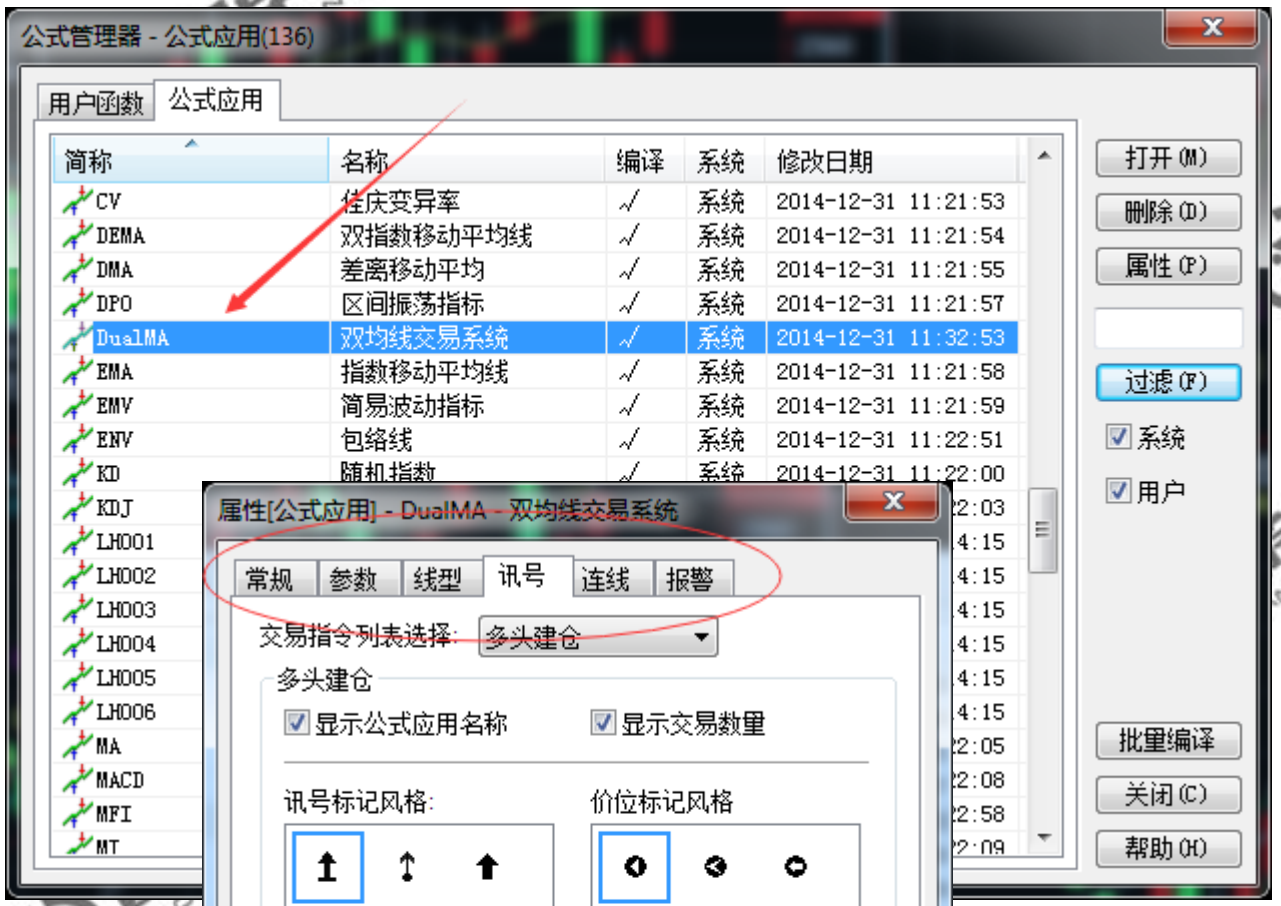
08月08日 08月11日 08月16日 08月19日

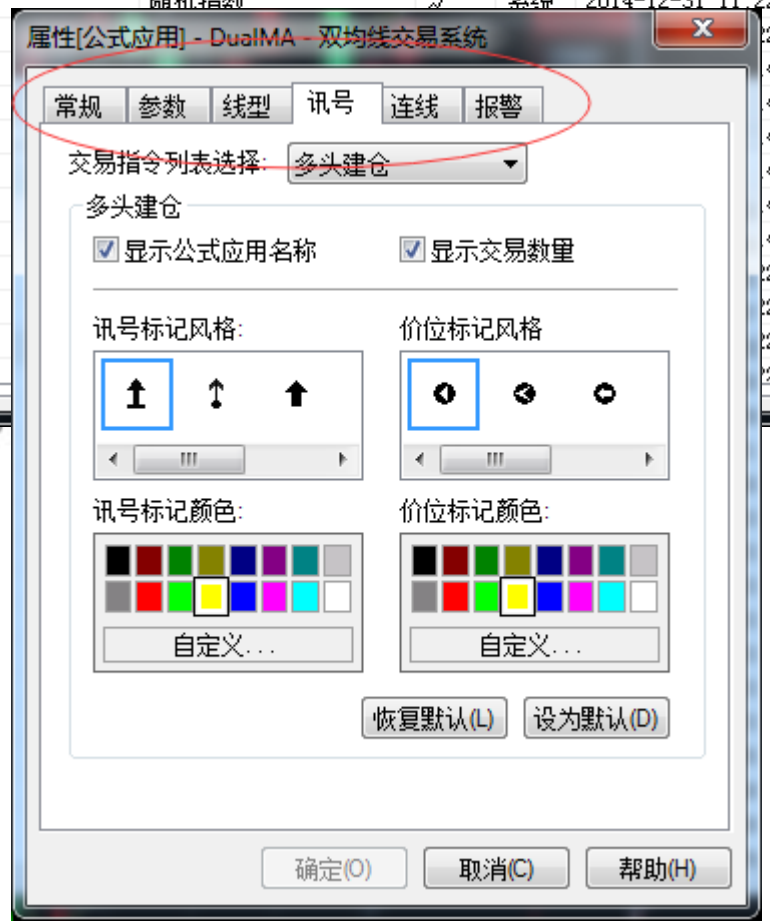
rb000 螺纹钢指数 1日线 2016/08/22 C=2565 4 0.16% O=2561 H=2580 L=2550

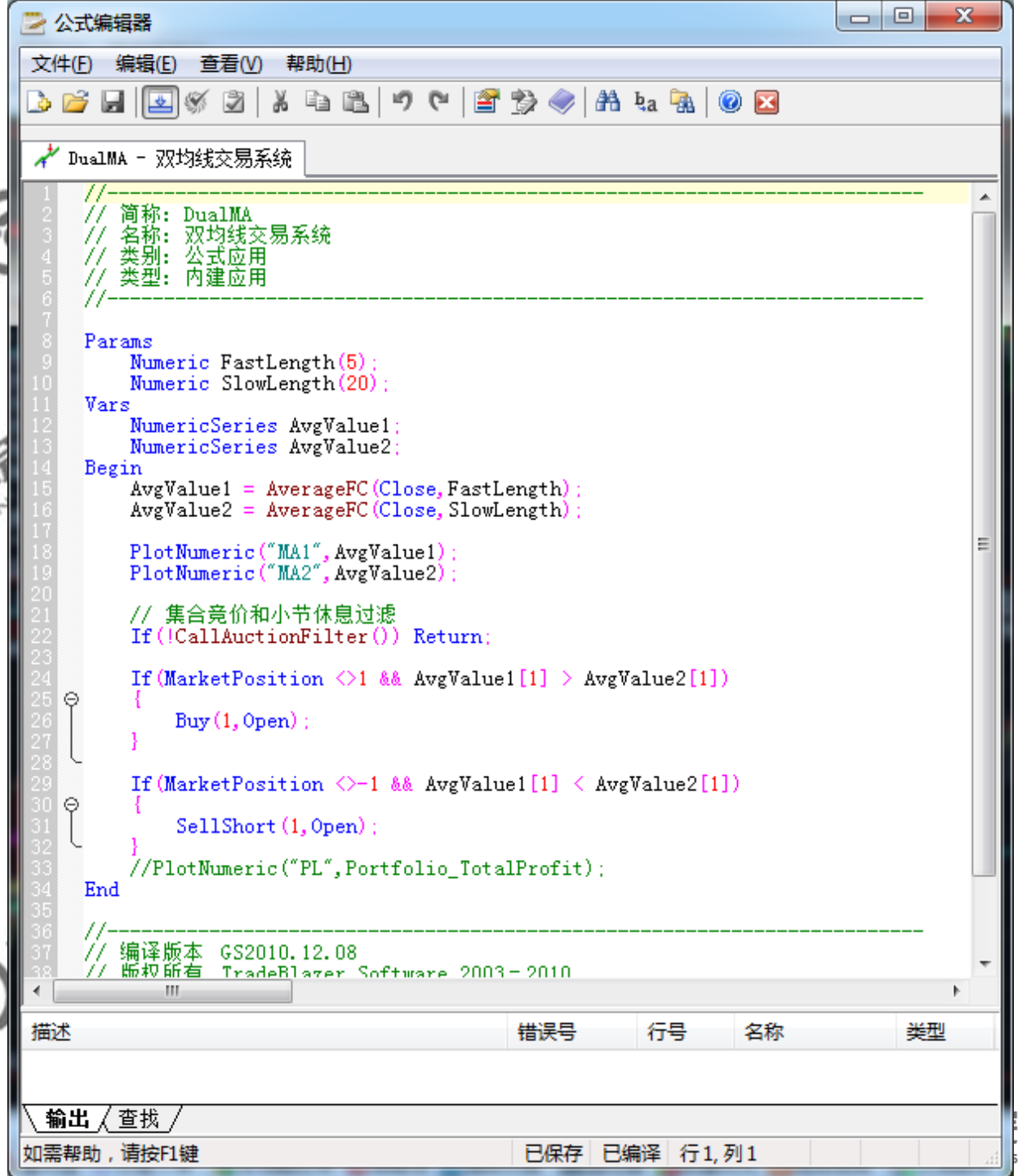
8月

(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








```
//-------------------------------------------------------------------
// 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
```

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		
总亏损	(83860.00)		
总盈利/总亏损	0.94		
交易手数	958		
盈利比率	30.58%		
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 | 时间范围[2014/01/02 09:00 - 2016/08/19 22:45]

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

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

这是一个简单的策略

策略 回测

已保存 编译运行

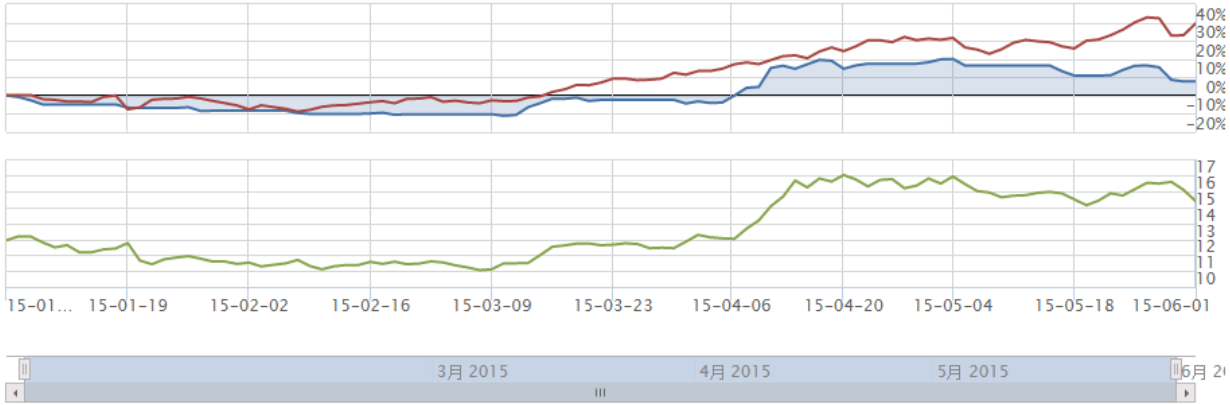
```python
# 定义一个全局变量，保存要操作的证券
# 000001(股票:平安银行)
security = '000001'
# 初始化此策略
# 设置我们要操作的证券池，这里我们只操作一支股票
set_universe([security])

# 每个单位时间(如果按天回撤,则每天调用一次,如果按分钟,则每分钟调用一次)调用一次
def handle_data(context, data):
    # 取得过去五年的平均价格
    average_price = data[security].mavg(5)
    # 取得当前价格
    current_price = data[security].price
    # 取得当先的现金
    cash = context.portfolio.cash

    # 如果当前价格高出五天平均价1%，而且至少可以买一股,则买入
    if current_price > 1.01*average_price and cash > current_price:
        # 计算可以买多少只股票
        number_of_shares = int(cash/current_price)
        # 调整股票单位为100股
        number_of_shares -= number_of_shares%100
        # 购买量大于0时，下单
        if number_of_shares > 0:
            # 买入股票
            order(security, +number_of_shares)
            # 记录这次买入
            log.info("Buying %s" % (security))
    # 如果当前价格低于五天平均价，则卖光这只股票
    elif current_price < average_price and context.portfolio.get_position(security).amount > 0:
        # 卖出所有股票,使这只股票的最终持有量为0
        order_target(security, 0)
        # 记录这次卖出
        log.info("Selling %s" % (security))
    # 画出当前的价格
    record(stock_price=data[security].price)
```

2015-01-01 到 2015-06-01 ¥ 10000 每天 运行回测

RETURNS	ALPHA	BETA	SHARPE	DRAWDOWN
7.54%	-0.11	0.46	0.34	11.29%



日志 运行时错误 更多

```
2015-01-05 - INFO - Buying 000001
2015-01-08 - INFO - Selling 000001
2015-01-19 - INFO - Buying 000001
2015-01-20 - INFO - Selling 000001
2015-01-26 - INFO - Buying 000001
2015-01-28 - INFO - Selling 000001
2015-02-06 - INFO - Buying 000001
2015-02-09 - INFO - Selling 000001
2015-02-16 - INFO - Buying 000001
2015-02-26 - INFO - Selling 000001
2015-03-10 - INFO - Buying 000001
2015-03-20 - INFO - Selling 000001
2015-03-31 - INFO - Buying 000001
2015-04-07 - INFO - Selling 000001
2015-04-08 - INFO - Buying 000001
2015-04-22 - INFO - Selling 000001
2015-04-29 - INFO - Buying 000001
2015-04-30 - INFO - Selling 000001
2015-05-04 - INFO - Buying 000001
2015-05-05 - INFO - Selling 000001
2015-05-14 - INFO - Buying 000001
2015-05-18 - INFO - Selling 000001
2015-05-21 - INFO - Buying 000001
2015-05-22 - INFO - Buying 000001
2015-05-29 - INFO - Selling 000001
```

```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(mongo
db = client.quant
coll_sec = db.securities
coll_trd = db.trades

# Read out all available
cursor = coll_sec.find({},
stock_list = [obj["trade_code"] for ob

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

# Get all valid trading dates
result = w.tdays("2015-01-01", "201
if result.ErrorCode != 0 or len(
    print("Error to get trad dates,
    sys.exit(-1)
trade_dates = [dt.strftime("%Y-%m-%d")
```


```python
        # I... packet buy-in request
        codes = ",".join([obj.trade_code fo... in accour
        amts = ",".join([str(obj.amount) ... obj in accour
        result = w.bktorder("%s 9:45:00" % dt, codes, "Buy
        if result.ErrorCode != 0:
            print("Error putting buying orders, aborted!")
            sys.exit(-1)
    else:
        # Adjust each stock's ... tion
        res1 = w.bktquery("...tal", "%s 9:40:00" % dt)
        res2 = w.bktquery(... osition", "%s 9:40:00" % dt)

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

三大类接口

数据接口

支持股票、...、期货、期权、债券、基金等各种标... 提供实时行情，历史行情，财务数据、... 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

海龟交易法则

PART TWO

一个交易系统的要素

- 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？

海龟交易系统是一个完整的交易系统

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
- 真实波动幅度 = Max（H-L, |H-PDC|, |PDC-L|）
- 其中：
- 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 = 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

系统1 以20日突破为基础的短期系统

- 如果上一次突破是一次赢利性突破，那么当前的入市信号将被忽略
- 55日突破点保障性信号

系统2 以55日突破为基础的长期系统

逐步建仓

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

直到头寸达到规模上限

止损

海龟止损标准

任何一笔交易的风险程度不超过账户的2%

- 价格变动的上限就是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

三种语言的实例

PART THREE

金字塔版本的实例

“多头海龟交易系统”


金字塔决策交易系统 - [免费登录] - [动态显示牌]
股票池 锁定 客服 策略库 公式 购买 账户 委托 网站 资讯
文件(F) 板块(M) 画面(P) 查看(V)
管理面板
新建公式 查找公式
交易系统
A.图表交易系统
1.指标交易
2.系统交易
03.早盘突破系统
04.顾比倒数线策略
05.三进三出交易法
06.K线形态交易系统
07.双向海龟交易系统
08.多头海龟交易系统
09.Hans123 【日内策略】
10.空中花园 【日内策略】
11.横盘突破 【日内策略】
15.Aberration 【长线策略】
16.Dual Thrust 【日内】
17.布林强盗系统 【中长】
18.金肯特纳交易系统
19.恒温器策略 【震荡】
20.闪灵交易者系统 Th
21.超级日内组合系统
B.后台交易系统
公式 指标组 高级
中金所 国内商品 期权
交易系统编辑 - (所属公式组：2.系统交易)
文件(F) 编辑(E) 查看(V) 插入(I) 调试(D) 帮助(H)
名称(N): 08.多头海龟交易系统
快捷码(Q): 08DTHGJYXT
用法注释(R)..
确定(K)
说明(D):
使用周期(P)
引入公式(O)..
副图(U) 主图(M)
加密(Y)
快速(I)..
费率设置(B)..
运行模式: 序列计算 仅刷最后一根K线 模式说明
编译公式(F)..
参数名 最小 最大 步长
调试信息:
开多 平多 开空 平空
颜色(T):
//该模型为简单示范模型，用户需根据自己交易经验，修改完善后再实际应用!!!
//变量申明
VARIABLE:DAYCOUNT=1,POSITIONCOUNT=1,SELLSIGN=0;//定义常数变量DAYCOUNT并初始化为1，定义常数变量POSITIONCOUNT并初始化为1，定
VARIABLE:ENTANDEXITSIGN=1,ENTPOINT=0,EXITPOINT=0;//定义常数变量ENTANDEXITSIGN并初始化为1，定义常数变量ENTPOINT并初始化为0
VARIABLE:N=0;//定义常数变量N并初始化为0
MA1:MA(C,5);//输出5日均价
MA3:MA(C,10);//输出10日均价
M:=MA(TR,20); //真实波幅的
BUYHHV:=HHV(H,20);//20
SELLLV:=LLV(L,10);//
SS:N,LINETHICKO;
//交易系统
IF BARPOS>=21 THEN BEGIN //如果从上市到现在的交易日天数大于等于21天，那么
IF BARPOS=21 THEN //如果从上市到现在的交易日天数等于21，那么
N:=M; //N=M
IF DAYCOUNT=6 OR BARPOS=21 THEN BEGIN{5天调整N值}
N:=(19*N+TR)/20;{计算N值}
DAYCOUNT:=2;
DAYCOUNT:=DAYCOUNT+1;
ENTPOINT:=ENTERBARS+1;
IF ENTPOINT=ENTANDEXITSIGN THEN BEGIN{说明
公式系统常见问题汇总
公式模型编写客服中心
公式模型编写调试教程
提高编写效率的注意事项
沪 3152.8 -3.39% (-110.72) 2934亿
深 10440 -4.02% (-437.26) 3419亿
300 3904.9 -2.87% (-115.41) 2189亿

```
//该模型为简单示范模型，用户需根据自己交易经验，修改完善后再实际应用！！！
//变量申明
VARIABLE:DAYCOUNT=1,POSITIONCOUNT=1,SELLSIGN=0;
VARIABLE:ENTANDEXITSIGN=1,ENTPOINT=0,EXITPOINT=0;
VARIABLE:N=0;
```

1

```
MA1:MA(C,5);   //输出5日均价
MA3:MA(C,10); //输出10日均价
M:=MA(TR,20); //真实波幅的20周期均值

BUYHHV:=HHV(H,20); //20日最高价
SELLLLV:=LLV(L,10);//10日最低价
```

```
SS:N,LINETHICK0;
```

2

```
//交易系统
IF BARPOS>=21 THEN BEGIN //如果从上市到现在的交易日天数大于等于21天，那么
    IF BARPOS=21 THEN    //如果从上市到现在的交易日天数等于21，那么
        N:=M;            //N=M
    IF DAYCOUNT=6 OR BARPOS=21 THEN BEGIN {5天调整N值}
        N:=(19*N+TR)/20; {计算N值}
        DAYCOUNT:=2;
    END
```

C：K线收盘价序列（CLOSE）
O/H/L/C：OPEN/HIGH/LOW/CLOSE

TR：真实波幅指标序列
MA：求数值序列的平均值
HHV：K线最高价序列的最大值
LLV：K线最低价序列的最小值

BARPOS：K线顺序位置
DAYCOUNT：循环累计天数
N：最新的波动性的值

```
DAYCOUNT:=DAYCOUNT+1;
ENTPOINT:=ENTERBARS+1;
IF ENTPOINT=ENTANDEXITSIGN THEN BEGIN {说明STOP指令买进头寸成功}
    POSITIONCOUNT:=POSITIONCOUNT+1;    {头寸计数}
    SELLSIGN:=TRUE; {开始以STOP卖出，如果达到指定的价格}
END
IF POSITIONCOUNT=1 THEN BEGIN {第一头寸}
    HOW:=CASH(0)*0.01/N; {波动性百分比决定头寸规模}
    开1:BUY(H>=BUYHHV,HOW,MARKET); {在20日新高STOP指令买进}
END
IF POSITIONCOUNT=2 THEN BEGIN {如到第二头寸}
    HOW:=CASH(0)*0.01/N; {波动性百分比决定头寸规模}
    开2:BUY(H>=ENTERPRICE+0.5*N,HOW,MARKET); {在上头寸(即第一头寸)+0.5个N以STOP指令买进}
END
IF POSITIONCOUNT=3 THEN BEGIN {如到第三头寸}
    HOW:=CASH(0)*0.01/N;
    开3:BUY(H>=ENTERPRICE+0.5*N,HOW,MARKET); {在上头寸(即第二头寸)+0.5个N以STOP指令买进}
END
IF POSITIONCOUNT=4 THEN BEGIN
    HOW:=CASH(0)*0.01/N;
    开4:BUY(H>=ENTERPRICE+0.5*N,HOW,MARKET);
END
```

1

2

3

ENTERBARS：上次开仓到当前的周期数|-1
ENTERPRICE：上次开仓价格
ENTERANDEXITSIGN：买入/卖出标志(=1)
SELLSIGN：是否（已有仓位）允许平仓
POSITIONCOUNT：累计开仓/加仓次数
BUY：参数1的条件满足时，多头买入开仓

从代码中看，最多允许开仓/加仓4次

“INTPART(.../100)*100”?

```
IF SELLSIGN=TRUE THEN BEGIN
    EXITPOINT:=EXITBARS+1;
    IF EXITPOINT=ENTANDEXITSIGN THEN BEGIN {说明卖出成功}
        POSITIONCOUNT:=1; {头寸计算复原}
        SELLSIGN:=FALSE;
    END
    IF ENTERPRICE-2*N THEN
        SELL(L<=SELLLLV,100%,MARKET); {退出盈利头寸}
    ELSE
        SELL(L<=ENTERPRICE-2*N,100%,MARKET); {退出亏损头寸}
END
END;
```

1

2

3

```
当前持仓:HOLDING,COLORGRAY,LINETHICK0;
当前资产:ASSET,NOAXIS,COLORGRAY;
```

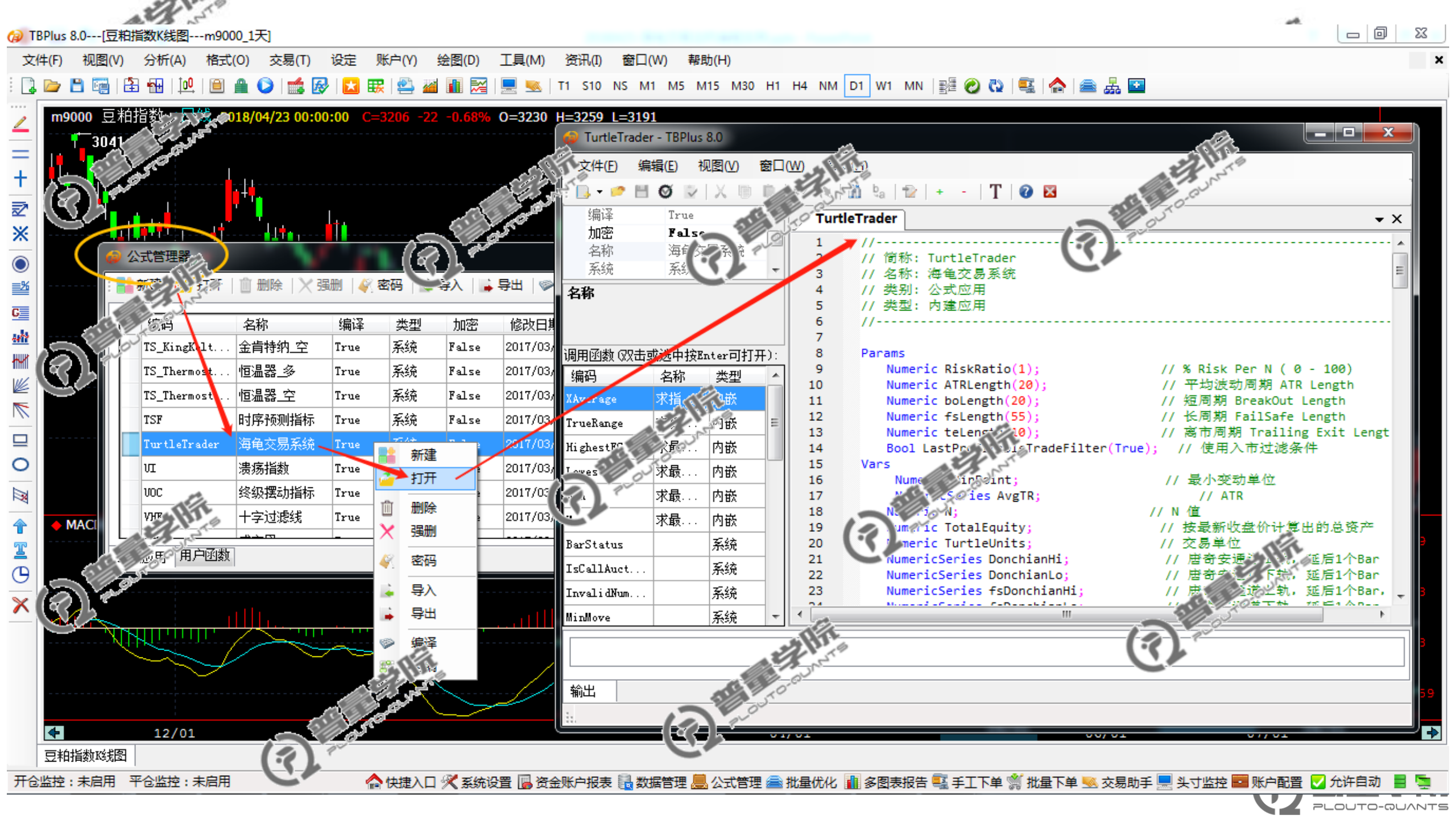
EXITBARS：上次平仓到当前的周期数|-1
SELL：参数1的条件满足时，多头卖出平仓

“L>ENTERPRICE-2*N”?


金字塔决策交易系统 - [免费登录] - [技术分析]
股票池 锁定 客服 策略库 公式 购买 账户 委托 网站 资讯 行情
文件(F) 板块(M) 画面(P) 查看(V) 交易(O) 分析(A) 工具(T) 窗口(W) 帮助(H)
50ETF(日线 复权) 17/12/26 开2.877 高2.897 低2.866 收2.893 量4927680 额14.2亿 振1.08% 涨(0.017)0.59%
08.多头海龟交易系统 MA1:2.883 MA3:2.865 SS:0.038 当前持仓:0 当前资产:1215265.625
3.038
2.655
-426100
-93
2.66
2017
10
11
12
日线
VOL(5,10,20) VOLUME:4927680 MA1:4777318.8 MA2:4816071.7 MA3:4919875.95
中金所 国内商品 期权 上海证券 深圳证券 恒指期货 新加坡市场 A股板块 外汇 伦敦LME CBOT粮油 NYME能源 日本期货 棕榈油 NYBOT商品 全球指数 芝加哥CME 其他金融期货
沪 3068.0 -0.11% (-3.53) 1579亿
深 10331 -0.75% (-78.28) 2359亿
300 3766.3 +0.15% (+5.48) 1118亿
17:25:02

交易开拓者(TB)版本的实例

"TurtleTrader - 海龟交易系统"


TBPlus 8.0---[豆粕指数K线图---m9000_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
m9000 豆粕指数 日线 2018/04/23 00:00:00 C=3206 -22 -0.68% O=3230 H=3259 L=3191
公式管理器
新建 打开 删除 强删 密码 导入 导出
编码 名称 编译 类型 加密 修改日期
TS_KingKelt... 金肯特纳_空 True 系统 False 2017/03/
TS_Thermost... 恒温器_多 True 系统 False 2017/03/
TS_Thermost... 恒温器_空 True 系统 False 2017/03/
TSF 时序预测指标 True 系统 False 2017/03/
TurtleTrader 海龟交易系统 True
UI 溃疡指数 True 2017/03/
UOC 终级摆动指标 True 2017/03/
VHF 十字过滤线 True 2017/03/
用户函数
新建
打开
删除
强删
密码
导入
导出
编译
TurtleTrader - TBPlus 8.0
文件(F) 编辑(E) 视图(V) 窗口(W)
编译 True
加密 False
名称 海龟交易系统
系统 系统
名称
调用函数(双击或选中按Enter可打开):
编码 名称 类型
XAverage 求指... 内嵌
TrueRange 内嵌
HighestBC 求最... 内嵌
Lowes 求最... 内嵌
求最... 内嵌
BarStatus 系统
IsCallAuct... 系统
InvalidNum... 系统
MinMove 系统
TurtleTrader
1 //----------
2 // 简称: TurtleTrader
3 // 名称: 海龟交易系统
4 // 类别: 公式应用
5 // 类型: 内建应用
6 //----------
7
8 Params
9 Numeric RiskRatio(1); // % Risk Per N (0 - 100)
10 Numeric ATRLength(20); // 平均波动周期 ATR Length
11 Numeric boLength(20); // 短周期 BreakOut Length
12 Numeric fsLength(55); // 长周期 FailSafe Length
13 Numeric teLength(10); // 离市周期 Trailing Exit Lengt
14 Bool LastPr... TradeFilter(True); // 使用入市过滤条件
15 Vars
16 Nume... inPoint; // 最小变动单位
17 N... ies AvgTR; // ATR
18 N... N; // N 值
19 ...meric TotalEquity; // 按最新收盘价计算出的总资产
20 ...meric TurtleUnits; // 交易单位
21 NumericSeries DonchianHi; // 唐奇安通... 延后1个Bar
22 NumericSeries DonchianLo; // 唐奇... 下轨, 延后1个Bar
23 NumericSeries fsDonchianHi; // 唐... 通上轨, 延后1个Bar,
输出
豆粕指数K线图
开仓监控：未启用 平仓监控：未启用
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

```
// 简称: TurtleTrader
// 名称: 海龟交易系统

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
```

1

改进版的唐奇安通道 => 海龟交易法

```
Begin
    // 集合竞价过滤
    If(BarStatus == 2 And IsCallAuctionTime) Return;

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

1

```
    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
```

2

```
    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);
```

3

BarStatus：当前K线位置状态|0/1/2
preEntryPrice：前一次开仓的价格
TrueRange：真实波幅序列
XAverage：求序列的指数平均
HighestFC：求N周期以来价格最高值
LowestFC：求N周期以来价格最低值

```
    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，这样能更接近真实情况，并能尽量保证成交
        myEntryPrice = min(high, DonchianHi + MinPoint);
        myEntryPrice = IIF(myEntryPrice < Open, Open, myEntryPrice); // 大跳空的时候用开盘价代替
        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，这样能更接近真实情况，并能尽量保证成交
        myEntryPrice = max(low, DonchianLo - MinPoint);
        myEntryPrice = IIF(myEntryPrice > Open, Open, myEntryPrice); // 大跳空的时候用开盘价代替
        preEntryPrice = myEntryPrice;
        SendOrderThisBar = True;
        SellShort(TurtleUnits, myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }
}
```

1

2

建仓1

MarketPosition：持仓方向|-1/0/1
Buy/Sell：多头开仓/平仓
SellShort/BuyToCover：空头开仓/平仓

```
// 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，这样能更接近真实情况，并能尽量保证成交
        myEntryPrice = min(high, fsDonchianHi + MinPoint);
        myEntryPrice = IIF(myEntryPrice < Open, Open, myEntryPrice); // 大跳空的时候用开盘价代替
        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，这样能更接近真实情况，并能尽量保证成交
        myEntryPrice = max(low, fsDonchianLo - MinPoint);
        myEntryPrice = IIF(myEntryPrice > Open, Open, myEntryPrice); // 大跳空的时候用开盘价代替
        preEntryPrice = myEntryPrice;
        SellShort(TurtleUnits, myEntryPrice);
        SendOrderThisBar = True;
        PreBreakoutFailure = False;
    }
}
```

1

2

建仓2

```
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); // 大跳空的时候用开盘价代替
        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) // 如果开盘就超过设定的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;
            }
        }
```

1 止盈

2 加仓

```
// 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); // 大跳空的时候用开盘价代替
    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); // 大跳空的时候用开盘价代替
        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) // 如果开盘就超过设定的1/2N,则直接用开盘价增仓
            {
                myEntryPrice = Open;
                preEntryPrice = myEntryPrice;
                SellShort(TurtleUnits, myEntryPrice);
                SendOrderThisBar = True;
            }
```

1

止损

上次突破不是赢利性突破的标志
(跌破2N止损的情况)

```
            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); // 大跳空的时候用开盘价代替
            BuyToCover(0, myExitPrice); // 数量用0的情况下将全部平仓
            PreBreakoutFailure = True;
        }
    }
}
End
```


TBPlus 8.0---[豆粕指数K线图---m9000_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
m9000 豆粕指数 1日线 2017/11/20 00:00:00 C=2858 -370 -11.46% O=2816 H=2871 L=2806 TurtleTrader
合约代码:m9000
2017/11/20[一]
时间 = 00:00:00
开 = 2816
高 = 2871
低 = 2806
收 = 2858
量 = 1753904
仓 = 2885082
涨跌 = 47.00
涨跌幅 = 1.67%
公式应用（TurtleTrader）
#N=28.476419
#preEntryPrice=2704.257502
#PreBreakoutFailure=False
#fsDonchianHi=2857
#fsDonchianLo=2675
信号（TurtleTrader）
Buy 200@2858.00
[200]
2675
2676.5
3224
3206
3150
3077
3003
2930
2856
2782
MACD(12,26,9) MACD 11.83 MACDAvg 7.13 MACDDiff 4.70 零线 0.00
80.06
62.58
55.77
50.77
21.48
0.00
-6.81
11/01
12/01
02/01
03/01
豆粕指数K线图
开仓监控：未启用 平仓监控：未启用
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

Python版本的实例

来源：https://blog.csdn.net/sbtgmz/article/details/51365265

Python资源列表

- Python官网
  - http://python.org
- 另一个常用的发行版本（Anaconda）
  - https://www.anaconda.com/download/
- 第三方的Python包目录
  - https://pypi.org
- Python扩展包（Windows下预编译好的）
  - https://www.lfd.uci.edu/~gohlke/pythonlibs/
- Python自学快速入门
  - https://www.liaoxuefeng.com (Python教程)

	A	B	C	D	E	F	G	H	I
1	index_code	date	open	close	low	high	volume	money	change
2	sh000300	5/8/2015	4515.55	4558.4	4445.59	4559.06	3.02E+08	4.52E+11	0.019756
3	sh000300	5/7/2015	4520.82	4470.09	4467.46	4546.34	2.98E+08	4.27E+11	-0.01828
4	sh000300	5/6/2015	4626.23	4553.33	4511.76	4700.91	3.76E+08	5.86E+11	-0.00947
5	sh000300	5/5/2015	4785.19	4596.84	4572.98	4785.19	4.6E+08	6.6E+11	-0.03987
6	sh000300	5/4/2015	4757.64	4787.74	4699.4	4795.92	3.78E+08	5.65E+11	0.007969
7	sh000300	4/30/2015	4788.41	4749.89	4749.48	4818.73	4E+08	6.13E+11	-0.00512
8	sh000300	4/29/2015	4722.9	4774.33	4695.79	4797.93	3.95E+08	6.02E+11	0.006848
9	sh000300	4/28/2015	4811.32	4741.86	4703.57	4839.08	6.09E+08	8.6E+11	-0.01367
10	sh000300	4/27/2015	4753.87	4807.59	4735.31	4810.49	5.13E+08	7.66E+11	0.022317
11	sh000300	4/24/2015	4682.63	4702.64	4618.32	4730.43	4.8E+08	7.21E+11	-0.00807
12	sh000300	4/23/2015	4764.48	4740.89	4693.97	4782.04	4.97E+08	7.52E+11	0.000228
13	sh000300	4/22/2015	4637.06	4739.81	4632.59	4740.93	5.16E+08	7.7E+11	0.026119
14	sh000300	4/21/2015	4520.46	4619.16	4504.56	4620.06	4.9E+08	6.85E+11	0.021504
15	sh000300	4/20/2015	4615.03	4521.92	4492.6	4671.17	6.86E+08	9.39E+11	-0.01615
16	sh000300	4/17/2015	4578.68	4596.14	4553.9	4630.27	5.53E+08	7.34E+11	0.018298
17	sh000300	4/16/2015	4355.49	4513.55	4335.63	4513.77	4.23E+08	5.65E+11	0.030371
18	sh000300	4/15/2015	4439.31	4380.51	4379.61	4476	4.72E+08	5.97E+11	-0.01299
19	sh000300	4/14/2015	4432.13	4438.18	4385.91	4474.35	4.34E+08	6.02E+11	0.00387
20	sh000300	4/13/2015	4394.05	4421.07	4361.59	4432.02	4.34E+08	5.99E+11	0.017643
21	sh000300	4/10/2015	4249.38	4344.42	4231.2	4351.69	3.48E+08	5.09E+11	0.019305
22	sh000300	4/9/2015	4316.96	4262.14	4212.61	4332.17	4.4E+08	6.5E+11	-0.00784
23	sh000300	4/8/2015	4277.45	4295.8	4204.83	4304.78	4.59E+08	6.58E+11	0.008394
24	sh000300	4/7/2015	4213.89	4260.04	4197.02	4260.47	4.15E+08	5.65E+11	0.02146
25	sh000300	4/3/2015	4104.67	4170.54	4092.38	4170.56	3.21E+08	4.67E+11	0.011094
26	sh000300	4/2/2015	4149.95	4124.78	4068.65	4156.84	3.39E+08	4.77E+11	0.000213
27	sh000300	4/1/2015	4057.5	4123.9	4046.94	4139.5	3.3E+08	4.67E+11	0.017945
28	sh000300	3/31/2015	4138.89	4051.2	4037.77	4166.02	4.36E+08	5.98E+11	-0.00905

http://bbs.pinggu.org/thread-3796364-1-1.html

```
import pandas as pd

# 导入上证指数的原始数据
index_data = pd.read_csv(r"C:\DataSet\all_trading_data\index data\sh000001.csv", parse_dates=['date'])

# 保留这几个需要的字段
index_data = index_data[["date", "high", "low", "close", "change"]]

# 对数据按照 [date] 交易日期从小到大排序
index_data.sort("date", inplace=True)

N1 = 20
N2 = 10

# 通过 rolling_max 方法计算最近 N1 个交易日的最高价
index_data['最近N1个交易日的最高点'] = pd.rolling_max(index_data['high'], N1)
#对于上市不足 N1 天的数据， 取上市至今的最高价
index_data['最近N1个交易日的最高点'].fillna(value=pd.expanding_max(index_data['high']), inplace=True)

# 通过相似的计算方法计算最近 N2 个交易日的最低价
index_data['最近N2个交易日的最低点'] = pd.rolling_min(index_data['low'], N1)
index_data['最近N2个交易日的最低点'].fillna(value=pd.expanding_min(index_data['low']), inplace=True)

# 当当天的 [close] > 昨天的 [最近N1个交易日的最高点] 时， 将 [收盘发出的信号] 设定为 1
buy_index = index_data[index_data['close'] > index_data['最近N1个交易日的最高点'].shift(1)].index
index_data.loc[buy_index, '收盘发出的信号'] = 1
```

1

```
# 当当天的 [close] < 昨天的 [最近N2个交易日的最低点] 时， 将 [收盘发出的信号] 设定为0
sell_index = index_data[index_data['close'] < index_data['最近N2个交易日的最低点'].shift(1)].index
index_data.loc[sell_index, '收盘发出的信号'] = 0

# 计算每天的仓位，当天持有上证指数时，仓位为 1，当天不持有上证指数时，仓位为 0
print '!!'*10
index_data['当天的仓位'] = index_data['收盘发出的信号'].shift(1)
print '@@'*10
index_data['当天的仓位'].fillna(method='ffill', inplace=True)

# 取1992年之后的数据，排除较早的数据
index_data = index_data[index_data['date'] >= pd.to_datetime('19930101')]
```

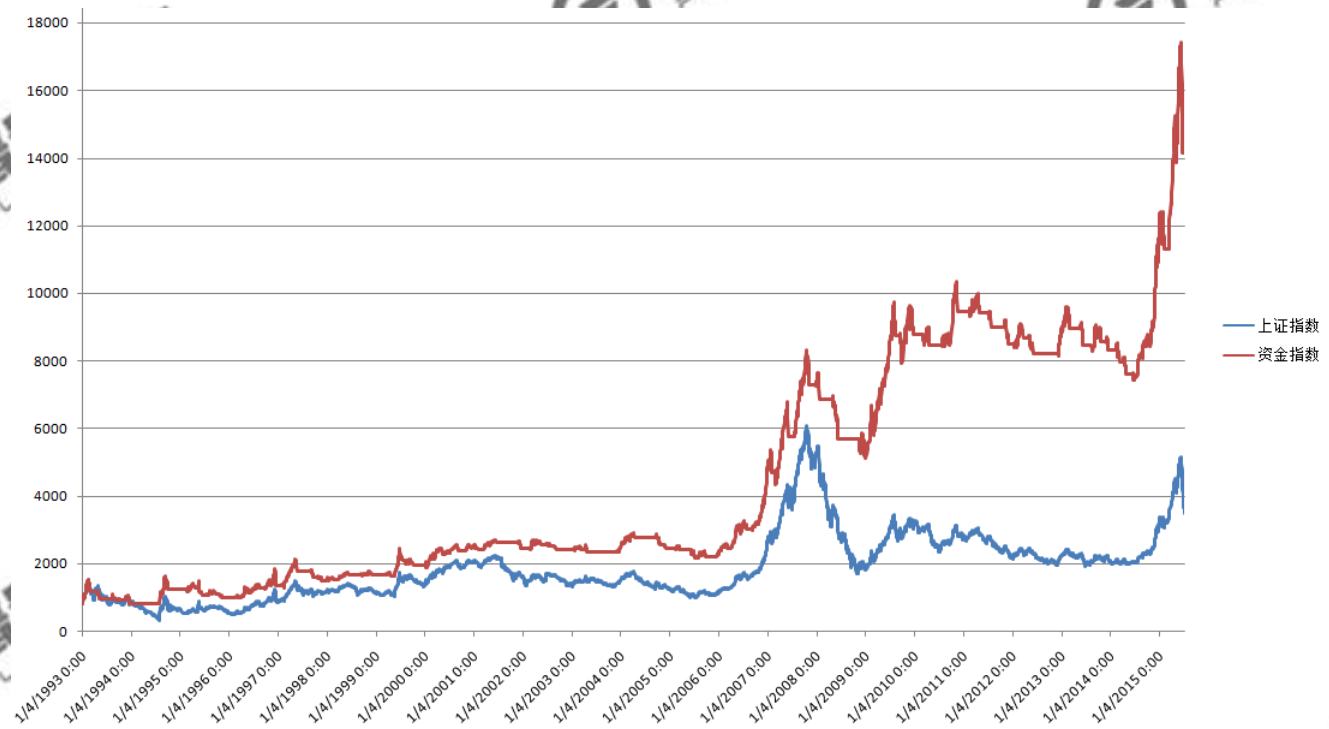
2

```
# 当仓位为1时，买入上证指数，当仓位为0时，空仓。计算从19930101至今的资金指数
index_data['资金指数'] = (index_data['change'] * index_data['当天的仓位'] + 1.0).cumprod()
initial_idx = index_data.iloc[0]['close'] / (1 + index_data.iloc[0]['change'])
index_data['资金指数'] *= initial_idx

# 输出数据到指定文件
index_data[['date', 'high', 'low', 'close', 'change', '最近N1个交易日的最高点', '最近N2个交易日的最低点', '当天的仓位',
'资金指数']].to_csv(r'C:\DataSet\testResult\turtle.csv', index=False, encoding='gbk')
```

3

```
index_data['海龟法则每日涨跌幅'] = index_data['change'] * index_data['当天的仓位']
year_rtn = index_data.set_index('date')[['change', '海龟法则每日涨跌幅']].\
    resample('A', how=lambda x: (x+1.0).prod() - 1.0) * 100
print year_rtn
print '运行结束'
```



http://bbs.pinggu.org/thread-3796364-1-1.html

1993-12-31	6.844014	5.676276
1994-12-31	-22.299112	51.277295
1995-12-31	-14.289904	-20.888385
1996-12-31	65.142538	35.896930
1997-12-31	30.215262	13.057451
1998-12-31	-3.969517	10.109529
1999-12-31	19.175024	16.272402
2000-12-31	51.727671	28.955647
2001-12-31	-20.617995	-1.966942
2002-12-31	-17.516723	-1.517241
2003-12-31	10.267005	0.807047
2004-12-31	-15.399722	0.284835
2005-12-31	-8.325306	-6.424061
2006-12-31	130.433397	110.611530
2007-12-31	96.659279	51.997364
2008-12-31	-65.394104	-30.321482
2009-12-31	79.982535	72.295600
2010-12-31	-14.313090	7.601559
2011-12-31	-21.675308	-10.197478
2012-12-31	3.169472	5.389615
2013-12-31	-6.749283	-7.238451
2014-12-31	52.869120	42.933098
2015-12-31	19.882029	19.041134

答疑热线

牛小秘

微信：niuxiaomi1

普量学院
PLOUTO-QUANTS