专业·创新·增值

金程教育
GOLDEN FUTURE

经典量化投资策略

Bush

CONTENTS

PROFESSIONAL · LEADING · VALUE-CREATING

专业来自101%的投入!

- 基于日期效应的量化投资
- 基于动量效应的量化投资

基于日期效应的量化投资策略

- 日期效应的基本原理及操盘策略
- 日期效应的应用举例：基于中国国情的二月效应
- 基于日期效应的量化投资策略的代码及注释
- 基于日期效应的量化投资策略的回测结果
- 模块复用：日期信息的获取

基本原理：

- 一月效应（日期效应）是从统计学角度分析股市走势的一种惯常现象，指一年中某个时间段的回报率往往是“正数”。最初出现日期效应是美国一月的股市收益率为正，因此得名为一月效应，然后其他国家的学者也陆续发现一月效应存在其他股市之中。
- 一月效应是相对的，如果市场是有效率及正常的话，原则上每个月的回报率平均也应是差不多。但根据统计学的验证结果显示，一月份的回报率往往是正数，而且会比其他月份为高；相反在十二月的股市回报率很多时会呈现负值，而学界称这种现象为“一月效应”。
- 中国的日期效中最简单的是二月效应，即春节所在月份，因此我们在2月初买入股票，3月初卖出股票。

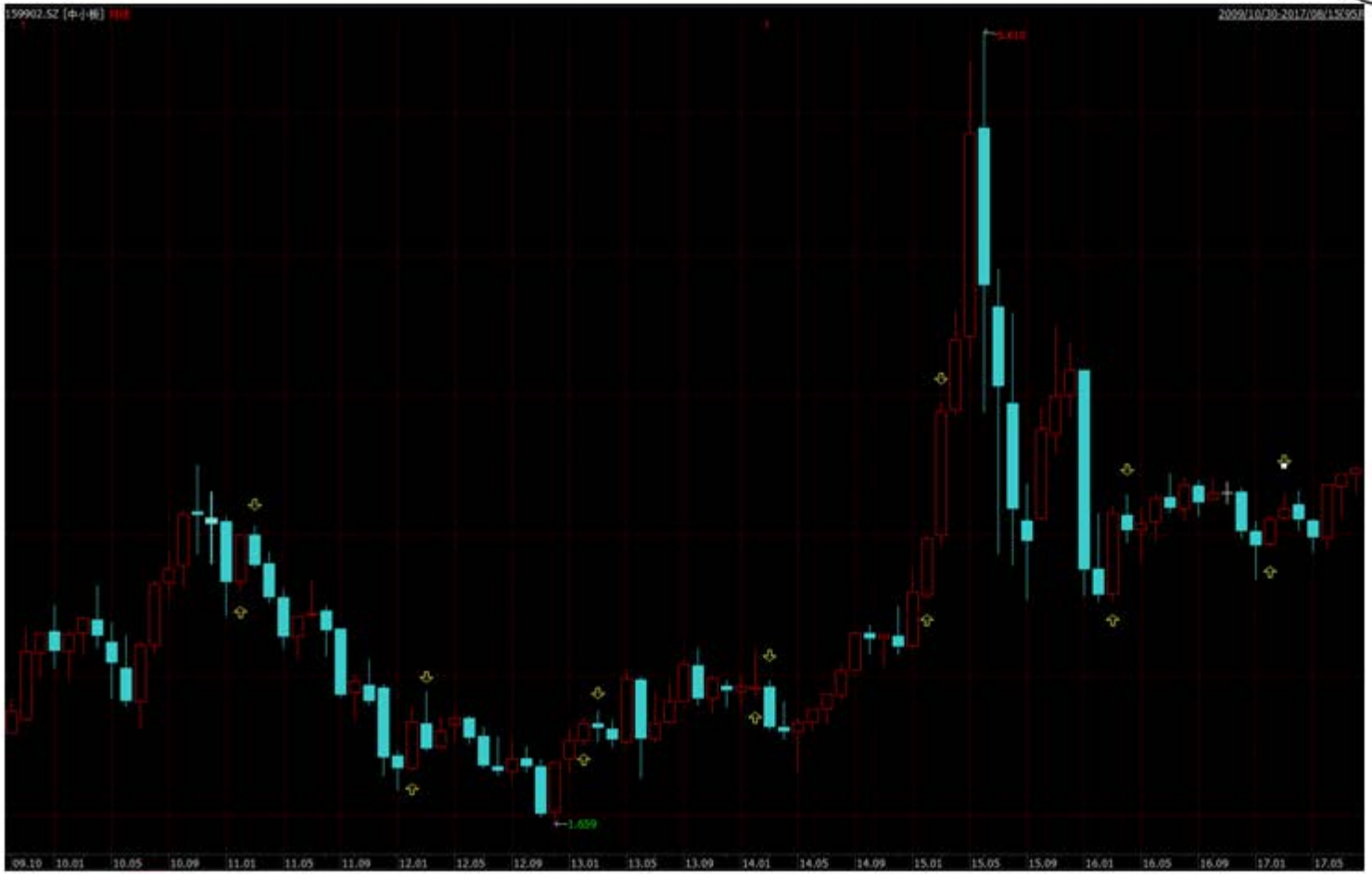
操作策略：

- 买入条件：2月第一个交易日买人股票
- 卖出条件：3月第一个交易日卖出股票

日期效应的基本原理：美国

- **January Effect or Turn-of -the year effect**
  - ✓ Definition: During the first five days of January, stock returns, especially for small firms are significantly higher than they are the rest of the year
  - ✓ Explanation:
    - ◆ Tax-loss selling
    - ◆ Window Dressing

基于中国国情的日期效应（二月效应）



基于日期效应的量化投资策略的代码及注释


```
#初始化回测环境
start = '20100101'  # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20170401'    # 回测结束时间
benchmark = '399005.ZICN'   # 策略参考标准
universe = set_universe('399005.ZICN', start)  # 证券池：可供选择的股票的范围; 注： 取start日的成分股防止survival bias
freq = 'd'          # 用日线回测的策略
refresh_rate = 1    # 每天调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数
buy_threshold = 0  # 买入阈值
sell_threshold = 0  # 卖出阈值

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
      pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
     timing(context)
```

基于日期效应的量化投资策略的代码及注释

金程教育 GOLDEN FUTURE

```
def timing_February(context):
    ##February算法在2月初买入，在3月初卖出              （操盘规则：预测）
    ##基于仓位管理的需要，每只股票最大投资额为100000元 （仓位管理：风控）


    #数据获取（通用部分）：投资者账户，可供投资股票，持仓数据，账户金额数据；注：由于使用系统提供的信号机制，不必获取价格数据

    today = context.current_date                                                    #获取当前日期

    account = context.get_account('fantasy_account')                               #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(exclude_halt=True)                     #获取当前除停牌外的所有可供投资股票（universe）

    security_position = account.get_positions()                                     #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
    cash = account.cash                                                             #获取股票账户可用于投资的现金额度

    #择时策略部分：获取当前时点的buylist
    buylist = []                                                                    #初始化购买股票列表
    selllist = []                                                                   #初始化卖出股票列表
    for sec in current_universe:                                                    #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票
        if today.month == 2 and sec not in security_position:                       #当前时间为2月，且无持仓，加入买入列表
            buylist.append(sec)
        elif today.month == 3 and sec in security_position:                         # 当前时间为3月，且有持仓，加入卖出列表
            selllist.append(sec)

    # 交易执行部分：卖出
    m = len(selllist)
    for sec in selllist[:m]:                                                         #全额出售卖出列表中的股票
       order_to(sec,0)                                                              #执行卖出指令后，会自动更新股票账户中的金额
   #交易执行部分：买入
    d = min(len(buylist), int(cash) // 100000)                                      #可以买入的股票数量，如果资金不够，只买入部分
    for sec in buylist[:d]:                                                         #买入buylist中前d只股票，d<=len(buylist)
        order(sec, 100000 / context.current_price(sec))                             #基于仓位管理的需要，每只股票最大投资额为100000元
```

```
def timing(context):
    timing_February(context)    #基础日期效应策略：2月效应策略
```

基于日期效应的量化投资策略的回测结果：HS300

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
4.5%	-0.5%	1.1%	0.07	0.13	6.7%	0.09	9.7%	7.74

回测详情　开始交易

专业来自101%的投入！

基于日期效应的量化投资策略的回测结果：中小板

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
5.3%	2.6%	1.7%	0.05	0.30	5.4%	-0.04	9.5%	6.06

回测详情　开始交易

专业来自101%的投入!

模块复用：日期信息的获取


```
today = context.current_date                                                    #获取当前日期

account = context.get_account('fantasy_account')                               #获取投资者的股票账户（fantasy_account）
current_universe = context.get_universe(exclude_halt=True)                      #获取当前除停牌外的所有可供投资股票（universe）
security_position = account.get_positions()                                     #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
cash = account.cash                                                             #获取股票账户可用于投资的现金额度

#择时策略部分：获取当前时点的buylist
buylist = []                                                                    #初始化购买股票列表
selllist = []                                                                   #初始化卖出股票列表
for sec in current_universe:                                                    #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票
    if today.month == 2 and sec not in security_position:                       #当前时间为2月，且无持仓，加入买入列表
        buylist.append(sec)
    elif today.month == 3 and sec in security_position:                         # 当前时间为3月，且有持仓，加入卖出列表
        selllist.append(sec)
```

基于动量效应的量化投资策略

- 动量效应的的基本原理及操盘策略
- 动量效应的应用举例
- 基于动量效应的量化投资策略的代码及注释
- 基于动量效应的量化投资策略的回测结果
- 模块复用：度量指标组的构建

动量效应的基本原理及操作步骤

- **基本原理：**
  - 股票收益率具有正相关性，即过去表现好（差）的股票未来依然表现好（差），这一现象被称为 “动量效应”。该理论的源自与物理学中动量概念的类比。我们把股票的价格类比为运动中的物体，股票价格的上涨或下跌则可视为物体的运动，依据惯性的原理，股票价格上涨，则其具有继续上涨的动能；同理，股票下跌，则可保持继续下跌的动能。
  - 基于行为金融学派的观点，动量效应的原因在于投资人在复杂的市场环境中无法完美地预期与判断，固有非理性的行为与随之而来的定价偏差，套利者受限于市场机制与风险承担能力，不一定能及时纠正偏差的价格。
- **操作策略：**
  - 买入条件：动量值排名在前20%，且未持有，则买入
  - 卖出条件：动量值排名不在前20%，且已持有，则卖出

➢ 动量效应产生的原因：

- 反应不足。投资者对信息的反应不足导致信息缓慢地反映在股价中，造成了股价变动的时滞，从而产生动量效应。
- 正反馈效应。前期价格上涨的股票和价格下跌的股票会在投资者买涨杀跌的正反馈交易下继续上涨或下跌，赢家继续赢，输家继续输，即羊群效应。
- 过度反应。投资者对私有信息的精确度过度自信，在得到信息后过度反应，导致股票价格持续上涨或下跌。

动量的计算方法：

- 作差法。当天价格减去一定时间周期以前的价格，计算公式如下:

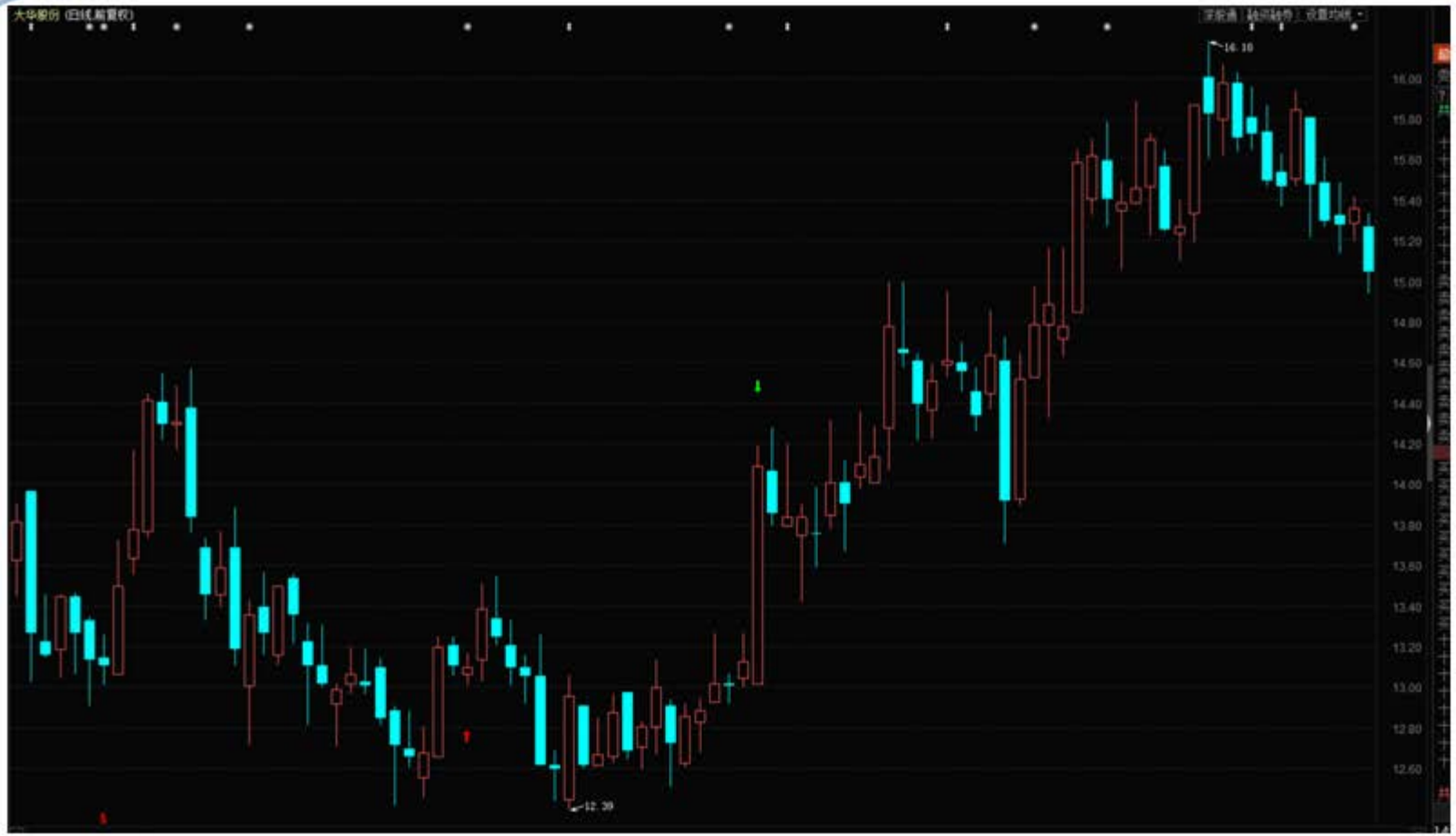
$$M_t = P_t - P_{t-n}$$

  - ✓ P_t为股票t期的价格，n表示动量的周期， P_{t-n}表示在t-n期的价格

- 作除法。t期的价格P_t减去n期以前的价格P_{t-n}，再除以P_{t-n}。做除法计算出的动量值是价格变化的比率（ROC：Rate of Change），因此计算结果更具有可比性，计算公式如下：

$$ROC_t = \frac{P_t - P_{t-n}}{P_t}$$

动量效应的应用举例



基于动量效应的量化投资策略的代码及注释


```
import pandas as pd #引入pandas类
#初始化回测环境
start = '20170510'  # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20170810'    # 回测结束时间
benchmark = 'HS300' # 策略参考标准
universe =  DynamicUniverse('HS300')  # 证券池：可供选择的股票的范围.
freq = 'd'          # 用日线回测的策略
refresh_rate = 1    # 每天调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数
Max_Position = 100000      #出于风险管理的目的，设定每只股票投资的最大仓位为100，000元
buy_percent = 0.2          #选择动量最值排名前20%的股票作为投资对象max_history_window = 250  #设置最长回测周期
max_history_window = 250  #设置最长回测周期
momentum_time_range =20 #计算动量的考察时间
#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    stock_selection(context)
```

基于动量效应的量化投资策略的代码及注释

金程教育 GOLDEN FUTURE

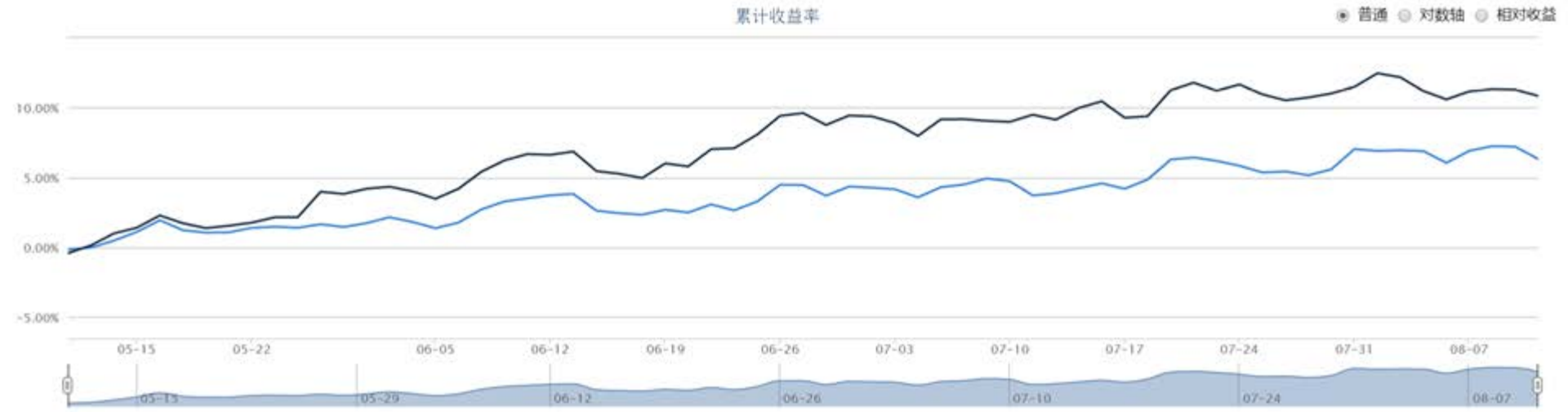
```
def stock_sellection_basic_momentum(context):
    ##买入动量值排名前20%的股票，其余卖出                （操盘规则：预测）
    ##基于仓位管理的需要，每只股票最大投资额为100000元 （仓位管理：风控）
    #数据获取（通用部分）：投资者账户，可供投资股票，持仓数据，账户金额数据：

    today = context.current_date                                                    #获取当前日期

    account = context.get_account('fantasy_account')                               #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(exclude_halt=True)                     #获取当前除停牌外的所有可供投资股票（universe）

    history = context.history(current_universe, 'closePrice', momentum_time_range) #获取股票收盘价数据用以计算动量，默认返回类型为dataframe
    security_position = account.get_positions()                                    #字典型数据，上一k线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
    cash = account.cash                                                            #获取股票账户可用于投资的现金额度
    #数据处理部分：计算动量（累计收益）
    momentum = {'symbol':[], 'moment':[]}                                              #momentum为字典型变量，以股票代码为key，动量值为value

    for sec in current_universe:                                                       #遍历证券池中所有股票
        if history[sec]['closePrice'][0]!=0:                                           #保证计算动量的首日收盘价不为空，去掉无效数据
            momentum['symbol'].append(sec)                                             #记录数据有效股票
            momentum['moment'].append(history[sec]['closePrice'][-1]/history[sec]['closePrice'][0])  #记录股票动量值（此处以除法计算动量取值）
    #数据处理部分：排序，取动量(累计收益)最高的20%
    momentum = pd.DataFrame(momentum).sort(columns='moment').reset_index()             #将证券池中股票根据动量值进行排序
    momentum = momentum[int(len(momentum)*(1-buy_percent)):len(momentum)]              #取动量值排名在前20%的股票
    momentum_list = momentum['symbol'].tolist()                                        #转换为list格式
    #选股策略部分：将动量值排名前20%的股票放入buylist,其余股票若有持仓，放入selllist
    buylist  = []                                                                  #初始化buylist
    selllist = []                                                                  #初始化selllist
    for sec in current_universe:
        #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票
        if sec in momentum_list and sec not in security_position:                  #动量值前20%股票，且无持仓，加入买入列表
            buylist.append(sec)
        elif sec not in momentum_list and sec in security_position:                #动量值排名不在前20%，且有持仓，加入卖出列表
            selllist.append(sec)
    # 交易执行部分：卖出
    m = len(selllist)
    for sec in selllist[:m]:                                                           #全额出售卖出列表中的股票
       order_to(sec,0)                                                                 #执行卖出指令后，会自动更新股票账户中的金额
    #交易执行部分：买入
    d = min(len(buylist), int(cash) // Max_Position)                                   #可以买入的股票数量，如果资金不够，只买入部分
    for sec in buylist[:d]:                                                            #买入buylist中前d只股票，d<=len(buylist)
        order(sec, Max_Position / context.current_price(sec))                          #基于仓位管理的需要，每只股票最大投资额为100000元
```

基于日期效应的量化投资策略的回测结果：HS300

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
26.6%	48.6%	-2.2%	0.59	3.22	8.3%	-2.31	1.4%	5.61

回测详情　开始交易



专业来自101%的投入！

模块复用：度量指标组的构建


```
momentum = {'symbol':[], 'moment':[]}                                                    #momentum为字典型变量，以股票代码为key，动量值为value
for sec in current_universe:                                                             #遍历证券池中所有股票
    if history[sec]['closePrice'][0]!=0:                                                 #保证计算动量的首日收盘价不为空，去掉无效数据
        momentum['symbol'].append(sec)                                                   #记录数据有效股票
        momentum['moment'].append(history[sec]['closePrice'][-1]/history[sec]['closePrice'][0])   #记录股票动量值（此处以除法计算动量取值）
```

CONTENTS

PROFESSIONAL · LEADING · VALUE-CREATING

专业来自101%的投入!

- 本杰明•格雷厄姆成长股投资法
- 本杰明•格雷厄姆的积极投资法
- 史蒂夫·路佛的价值投资策略
- 惠特尼•乔治小型价值股投资法

本杰明•格雷厄姆成长股投资法

- 本杰明•格雷厄姆的简介
- 成长股投资法的基本原理及操盘策略
- 成长股投资法的应用举例
- 成长股投资法的量化投资策略的代码及注释
- 成长股投资者法的量化投资策略的回测结果
- 模块复用：不同来源因子的合并计算

本杰明•格雷厄姆的简介

- 本杰明·格雷厄姆（BenjaminGraham，1894~1976年），美国经济学家和投资思想家，投资大师，“现代证券分析之父”，价值投资理论奠基人。
- 格雷厄姆在投资界的地位，相当于物理学界的爱因斯坦，生物学界的达尔文。作为一代宗师，他的证券分析学说和思想在投资领域产生了极为巨大的震动，影响了几乎三代重要的投资者。如今活跃在华尔街的数十位上亿的投资管理人都自称为格雷厄姆的信徒，因此，享有“华尔街教父”的美誉。
- 格雷厄姆不仅是沃伦·巴菲特就读哥伦比亚大学经济学院的研究生导师，而且被巴菲特膜拜为其一生的“精神导师”，“血管里流淌的血液80%来自于格雷厄姆”。青年时代的巴菲特师从格雷厄姆，在其投资公司工作跟随其学习投资，终于初步领悟价值投资的真谛。两年时间就把个人资产从9 800 美元增长到17 400 美元。

➢ 基本原理：

- 价值投资之父”格雷厄姆在其畅销书《聪明的投资者》中给出了一个对成长股内在价值进行估值的简单公式，但其计算出的数据，非常接近于一些更加复杂的数学计算所得。
- 成长价值 = 当期(正常)利润 × (8.5 + 两倍的预期年增长率)。
  - ✓ 当期正常利润 = 稀释每股收益 * 流通股的数量
  - ✓ 预期利润 = 5年增长率；（注：5年收益增长率（Five-year earnings growth）。计算方法：5年收益增长率= 5年收益关于时间（年）进行线性回归的回归系数/5年收益均值的绝对值。）
- 当成长价值>股票流通市值，说明股票被低估，所以应买入，反之若成长价值<股票的流通市值，说明股票被高估，应卖出股票。

➢ 操作策略：

- 买入条件：成长价值>股票流通市值，买入
- 卖出条件：成长价值<股票流通市值，卖出

- The **basic EPS** calculation does not consider the effects of any dilutive securities in the computation of EPS.

$$\text{basic EPS} = \frac{NI - div_{\text{preferred stock}}}{\text{weighted average number of common shares outstanding}}$$

- Weighted average number of common share outstanding
  - New issue, repurchase is **weighted by time (days or months);**

- Johnson company has net income of $10,000 and paid $1,000 cash dividend to its preferred shareholders and $1,750 cash dividend to its common shareholders. At the beginning of the year, there were 10,000 shares of common stock outstanding. 2,000 new shares were issued on July 1. what is johnson's basic EPS?
- **Answer：**
  - Weighted average shares = 10,000 × (12/12) + 2,000 ×(6/12) = 11,000
  - BEPS = ($10,000 - $1,000)/11,000 = $0.82

- **In the case of diluted EPS, if there are dilutive securities, then the numerator must be adjusted as follows:**
  - If convertible preferred stock is dilutive (meaning EPS will fall if it is converted to common stock), the convertible preferred dividends must be added to earnings available to common shareholders.
  - If convertible bonds are dilutive, then the bonds' after-tax interest expense is not considered an interest expense for diluted EPS. Hence, interest expense multiplied by (1 - the tax rate) must be added back to the numerator.

$$\text{diluted EPS} = \frac{\text{adjusted income available for common shares}}{\text{weighted avg. common \& potential common shares out}}$$

$$= \frac{\begin{bmatrix} \text{NI} \\ -\ \text{div}_{\text{preferred}} \end{bmatrix} + \begin{bmatrix} \text{div}_{\substack{\text{convertible} \\ \text{preferred}}} \end{bmatrix} + \begin{bmatrix} \text{interest}_{\substack{\text{convertible} \\ \text{debt}}} \end{bmatrix} (1-t)}{WACSO + \begin{bmatrix} \text{shares}_{\substack{\text{conversion of} \\ \text{conv. pfd. shares}}} \end{bmatrix} + \begin{bmatrix} \text{shares}_{\substack{\text{conversion of} \\ \text{conv. debt}}} \end{bmatrix} + \begin{bmatrix} \text{shares}_{\substack{\text{issuable from} \\ \text{stock opt.}}} \end{bmatrix}}$$

➢ **EPS with convertible debt**

During 2006 GF corp. reported net income of $115,600 and had 200,000 shares of common stock outstanding for the entire year. GF also had 1,000 shares of 10%, par $100 preferred stock outstanding during 2006. During 2005, GF issued 600, $1,000 par, 7% bonds for $600,000 (issued at par). Each of these bonds is convertible to 100 shares of common stock. The tax rate is 40%.

Compute the 2006 basic and diluted EPS.

➢ **Answer：**

- BEPS = ($115,600 - $100 × 1000 × 10%)/ 200,000 = 52.8 Cents
- Adjusted income available for common shares = $115,600 – pref div $10,000 + int saving $600,000 × 7% × (1-40%) = $130,800
- Weighted average common shares and potential shares = 200,000 + 600 × 100 = 260,000
- DEPS = $130,800 / 260,000 = 50.3 Cents

成长股投资法的应用举例

金程教育 GOLDEN FUTURE

secID	stock_name	tradeDate	DilutedEPS	EGRO	MarketValue	Value	under_value
000651	格力电器	2015/9/23	1.9	0.2122	98258835675	1E+11	3024580000
000651	格力电器	2015/10/15	1.9	0.2122	1.04232E+11	1E+11	-2948601000

成长股投资法的量化投资策略的代码及注释

金程教育 GOLDEN FUTURE

```
#加载库函数
import datetime                          #加载时间函数库
#初始化回测环境
start = '20150601'                       # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20160601'                         # 回测结束时间
benchmark = 'HS300'                      # 策略参考标准
universe =  DynamicUniverse('HS300')    # 证券池：可供选择的股票的范围.
freq = 'd'                               # 用日线回测的策略
refresh_rate = 10                        # 每10日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position = 1000000                   #每只股票的买入限额为1,000，000元

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    buylist = stock_sellection_Graham_growth_stock(context)  # 基于本杰明•格雷厄姆成长股思想的选股策略
    trading(buylist,context)                                 # 交易
```

成长股投资法的量化投资策略的代码及注释

金程教育 GOLDEN FUTURE

```
def stock_sellection_Graham_growth_stock(context):

    """
    操盘规则(预测)：成长价值=当期(正常)利润×(8.5 + 两倍的预期年增长率)
                    成长价值>市值，买入，反之，卖出
    """

    #数据获取（通用部分）：投资者账户，可供投资股票
    previous_date = context.previous_date                                          #获取上一交易日的时间
    previous_date = previous_date.strftime('%Y%m%d')                               #将日期格式调整为%Y%m%d形式（20150227），方便读取因子数据
    account = context.get_account('fantasy_account')                              #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)  #获取当前除停牌外的所有可供投资股票（universe）

    #数据获取（策略需要数据）：EGRO数据（5年收益增长率），收盘价，DilutedEPS（稀释每股收益）
    GGV_EPS_EGRO = DataAPI.MktStockFactorsOneDayGet(tradeDate=previous_date,secID=current_universe,field=['secID','DilutedEPS','EGRO'],pandas="1")     #获取5年增长率和稀释每股收益
    GGV_closePrice = DataAPI.MktEqudGet(tradeDate=previous_date,secID=current_universe,isOpen="",field=u"secID,closePrice,negMarketValue",pandas="1") #获取收盘价和流动股总市值数据

    #数据处理部分（数据结构部分）：设定索引
    GGV_EPS_EGRO = GGV_EPS_EGRO.set_index(['secID'])                               #设置EPS和EGRO的索引
    GGV_closePrice = GGV_closePrice.set_index(['secID'])                           #设置收盘价和流通股市值数据的索引
    #数据处理部分（策略计算部分）：计算股票的成长价值
    GGV_closePrice['Maketable_Volume'] = GGV_closePrice['negMarketValue'] / GGV_closePrice['closePrice']                #计算流动股的数量
    GGV_EPS_EGRO['Value'] = GGV_closePrice['Maketable_Volume']*GGV_EPS_EGRO['DilutedEPS']*(8.5+2*GGV_EPS_EGRO['EGRO'])  #计算股票的成长价值
    under_value = GGV_EPS_EGRO['Value'] - GGV_closePrice['negMarketValue']                                              #计算股票的低估价值

    #策略构建部分：用被低估股票，生成buylist
    buylist =  under_value[under_value>0].index                                    #选择被低估股票,生成buylist
    return  list(buylist)                                                          #返回buylist
```

成长股投资法的量化投资策略的代码及注释

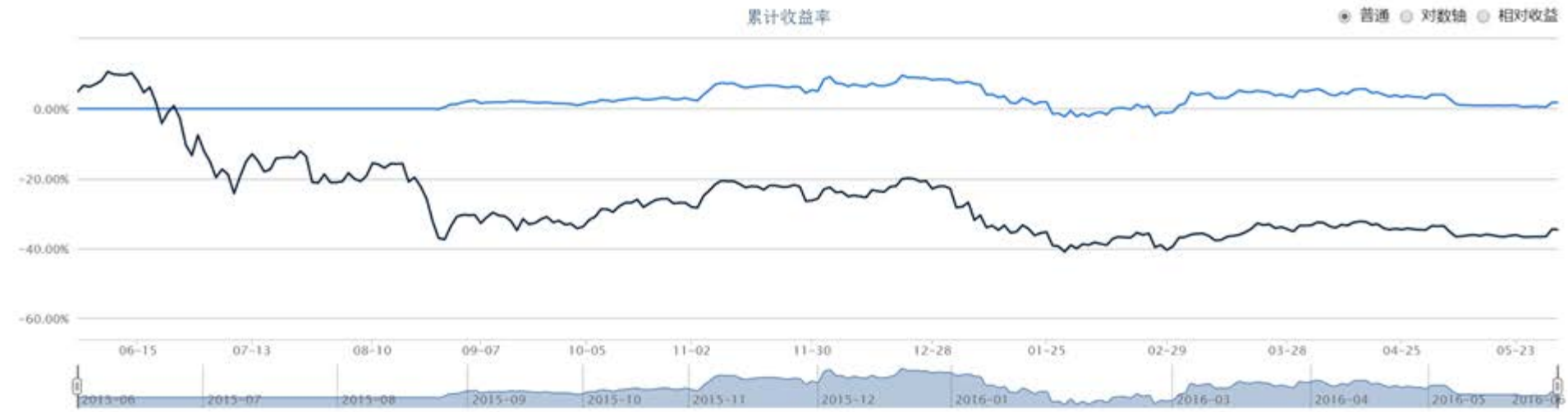

```
def trading(buylist,context):

 """
 仓位管理（固定金额）：单只股票的买入限额为1000，000元
 """

 #数据获取（通用部分）：投资者账户，可供投资股票
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

 #交易执行部分：卖出
 for sec in current_universe:
    if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 d = min(len(buylist), int(cash) // Max_Position)                                    #可以买入的股票数量，如果资金不够，只买入部分
 for sec in buylist[:d]:                                                             #买入buylist中前d只股票，d<=len(buylist)
    if sec not in security_position:
        order(sec, Max_Position / context.current_price(sec))                        #基于仓位管理的需要，每只股票最大投资额为100000元
```

成长股投资者法的量化投资策略的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
1.8%	-35.0%	3.8%	0.15	-0.16	11.3%	1.08	10.8%	1.40

回测详情　开始交易



专业来自101%的投入!

模块复用：不同来源因子的合并计算

金程教育 GOLDEN FUTURE

```
#模块复用： 不同来源因子的合并

universe = set_universe('HS300')
    #数据获取（策略需要数据）：EGRO数据（5年收益增长率），收盘价，DilutedEPS（稀释每股收益）
GGV_EPS_EGRO = DataAPI.MktStockFactorsOneDayGet(tradeDate=u"20150227",secID=universe,field=['secID','DilutedEPS','EGRO'],pandas="1")     #获取5年增长率和稀释每股收益
GGV_closePrice = DataAPI.MktEqudGet(tradeDate=u"20150227",secID=universe,isOpen="",field=u"secID,closePrice,negMarketValue",pandas="1") #获取收盘价和流动股总市值数据
    #数据处理部分（数据结构部分）： 设定索引
GGV_EPS_EGRO = GGV_EPS_EGRO.set_index(['secID'])                                                    #设置EPS和EGRO的索引
GGV_closePrice = GGV_closePrice.set_index(['secID'])                                                #设置收盘价和流通股市值数据的索引
    #数据处理部分（策略计算部分）： 计算股票的成长价值
GGV_closePrice['Maketable_Volume'] = GGV_closePrice['negMarketValue'] / GGV_closePrice['closePrice']                #计算流动股的数量
GGV_EPS_EGRO['Value'] = GGV_closePrice['Maketable_Volume']*GGV_EPS_EGRO['DilutedEPS']*(8.5+2*GGV_EPS_EGRO['EGRO'])  #计算股票的成长价值
GGV_EPS_EGRO['Maketable_Volume'] = GGV_closePrice['Maketable_Volume']
GGV_EPS_EGRO
```

模块复用：不同来源因子的合并计算

	DilutedEPS	EGRO	Value	Maketable_Volume
secID				
000001.XSHE	1.3700	0.1809	1.194242e+11	9.836712e+09
000002.XSHE	0.5900	0.1780	5.071778e+10	9.706677e+09
000009.XSHE	0.1731	0.2599	2.325861e+09	1.489669e+09
000060.XSHE	0.2000	-0.2905	3.263803e+09	2.060742e+09
000063.XSHE	0.5300	0.5522	1.425896e+10	2.801185e+09
000069.XSHE	0.3870	0.1178	1.052224e+10	3.112464e+09
000100.XSHE	0.2450	0.2286	1.794551e+10	8.177444e+09
000157.XSHE	0.1300	-0.5661	5.996869e+09	6.260995e+09
000402.XSHE	0.1900	0.0304	4.856858e+09	2.985984e+09
000423.XSHE	1.4115	0.1506	8.121089e+09	6.537196e+08
000425.XSHE	0.3700	-0.5782	6.402927e+09	2.356502e+09
000538.XSHE	1.8700	0.1931	1.730457e+10	1.041365e+09
000568.XSHE	0.8890	-0.2407	9.983243e+09	1.400462e+09
000623.XSHE	1.0220	0.1300	7.064060e+09	7.890407e+08
000625.XSHE	1.1700	0.5213	3.782111e+10	3.387518e+09

本杰明•格雷厄姆积极投资法

- 本杰明•格雷厄姆的简介
- 积极投资法的基本原理及操盘策略
- 积极投资法的应用举例
- 积极投资法的量化投资策略的代码及注释
- 积极投资法的量化投资策略的回测结果
- 模块复用：选择满足条件的因子

本杰明•格雷厄姆的简介

- 本杰明·格雷厄姆（BenjaminGraham，1894~1976年），美国经济学家和投资思想家，投资大师，“现代证券分析之父”，价值投资理论奠基人。
- 格雷厄姆在投资界的地位，相当于物理学界的爱因斯坦，生物学界的达尔文。作为一代宗师，他的证券分析学说和思想在投资领域产生了极为巨大的震动，影响了几乎三代重要的投资者。如今活跃在华尔街的数十位上亿的投资管理人都自称为格雷厄姆的信徒，因此，享有“华尔街教父”的美誉。
- 格雷厄姆不仅是沃伦·巴菲特就读哥伦比亚大学经济学院的研究生导师，而且被巴菲特膜拜为其一生的“精神导师”，“血管里流淌的血液80%来自于格雷厄姆”。青年时代的巴菲特师从格雷厄姆，在其投资公司工作跟随其学习投资，终于初步领悟价值投资的真谛。两年时间就把个人资产从9 800 美元增长到17 400 美元。

积极投资法基本原理及操作步骤

- **基本原理：**
  - 本杰明.格雷厄姆在《聪明的投资者》“积极型投资者”的六条选股标准，再加入对PE的限制（书中的基础准则）之后，构成了本杰明·格雷厄姆对积极投资者的建议。
    - ✓ 股票的市盈率低于市场水平
    - ✓ 股票的市净率小于1.2
    - ✓ 企业的流动比率要大于1.5
    - ✓ 企业总借款不超过流动资产的1.1倍
    - ✓ 最近5年净利润均大于零
    - ✓ 利润增长处于较高水平
  - 根据中国市场的实践，对具体参数进行修改：
    - ✓ 股票的市盈率高于市场水平
    - ✓ 股票的市净率小于1.2
    - ✓ 企业的流动比率要大于1.5
    - ✓ 企业总借款不超过流动资产的1.1倍
    - ✓ 利润增长处于较高水平：大于80%的股票
- **操作策略：**
  - 买入条件：满足中国市场6条规则，则买入
  - 卖出条件：不满足中国市场6条规则，则卖出

➢ Two methods for

- The method of **historical average EPS**
  - ✓ Normal EPS is calculated as average EPS over the **most recent full cycle**
  - ✓ one of several possible statistical approaches to the problem of **cyclical earnings**
  - ✓ the method does not account for **changes in the business's size** however
- The method of **average ROE**
  - ✓ Normal EPS is calculated as the average return on equity from the **most recent full cycle, multiplied by current book value per share**
  - ✓ by using recent book value per share, reflects more accurately the effect on EPS of **growth or shrinkage in the company's size**
  - ✓ For that reason, the method of average ROE is **sometimes preferred**

积极投资法基本原理及操作步骤：市盈率（EPS调整）

- Using the data in the following figure, calculate normalized earnings using the method of historical average EPS and the method of average return on equity for Magnolia Enterprises.
- Data for Magnolia Enterprises [amounts in Canadian dollars (C$)]

Year	2006	2007	2008	2009
EPS	4.20	3.75	4.75	4.30
BVPS	26.02	27.78	29.25	32.29
ROE	14.0%	12.0%	16.0%	14.0%

- Answer:

$$\text{Normalized earnings (average EPS approach)} = \frac{4.20+3.75+4.75+4.30}{4} = 4.25$$

$$\text{Average ROE} = \frac{0.14+0.12+0.16+0.14}{4} = 0.14 = 14.00\%$$

- Normalized earnings are C$4.25 based on the method of historical average EPS and C$4.52 based on the method average return on equity.

$$\text{Normalized earnings (average ROE approach)} = \text{Average ROE} \times \text{BVPS}_{2009}$$
$$= 0.14 \times 32.29 = 4.52$$

➢ Advantages & Disadvantages

- Advantages:
  - ✓Earning power is the primary determinant of value;
  - ✓P/E ratio is popular;
  - ✓P/E differences are related to long-run average stock returns
- Disadvantage:
  - ✓Earnings might be negative;
  - ✓volatile earning;
  - ✓management discretion distorts

- The measure of value in the P/B ratio, book value per share, is a stock or level variable coming from the balance sheet.
- Book value per share attempts to represent the investment that common shareholders have made in the company, on a per-share basis.
- The **computation of book value**:
  - Common shareholders' equity = (Shareholders' equity) - (the total value of equity claims that are senior to common stock)
  - (Common shareholders' equity)/(number of common stock shares outstanding) = book value per share

➢ **Advantages**:

- BV almost always>0,
- BV more stable than EPS.
- Measures NAV per share, more fit for valuing companies composed chiefly of liquid assets, such as finance institutions.
- Differences in P/Bs may be related to differences in long-run average returns, according to empirical research.

➢ **Disadvantages**:

- Size differences cause misleading comparisons.
- Influenced by accounting choices.
- BV≠MV due to inflation/ technology.
- Other assets besides those recognized in accounting may be critical.

积极投资法的应用举例

金程教育 GOLDEN FUTURE

secID	tradeDate	PE	PB	CurrentRatio	LDtWC	NetProfitGrowRate
000560	2016/5/17	248.1074	2.9982	2.1253	0.3568	2.2678
000560	2016/5/31	249.152	3.0108	2.1253	0.3568	2.2678

积极投资法的量化投资策略的代码及注释


```
#加载库函数
import datetime                        #加载时间函数库
#初始化回测环境
start = '20160101'                     # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20170101'                       # 回测结束时间
benchmark = '000002.ZICN'              # 策略参考标准为A股指数
universe = set_universe('A')           # 证券池：可供选择的股票的范围为A股.（由于选股条件比较苛刻，所以为了保证有充足的可供选择股票，因此以整个A股作为投资范围）
freq = 'd'                             # 用日线回测的策略
refresh_rate = 10                      # 每10日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position = 1000000                 #每只股票的买入限额为1,000，000元

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    buylist = stock_sellection_Graham_positove_stock(context)  # 基于本杰明·格雷厄姆积极选股思想的选股策略
    trading(buylist,context)                                   # 基于固定投资额的仓位管理策略
```

积极投资法的量化投资策略的代码及注释

```
def stock_sellection_Graham_positove_stock(context):

    """
    操盘规则(预测)：
    * 股票市盈率高于市场平均水平
    * 股票的市净率低于3
    * 企业的流动比率大于1.1
    * 企业的长期负债与营运资金(流动资产-流动负债)比率不超过5
    * 净利润增长率处于较高水平
    """

    #数据获取（通用部分）：投资者账户，可供投资股票
    previous_date = context.previous_date                                          #获取上一交易日的时间
    previous_date = previous_date.strftime('%Y%m%d')                               #将日期格式调整为%Y%m%d形式（20150227），方便读取因子数据
    account = context.get_account('fantasy_account')                              #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)  #获取当前除停牌外的所有可供投资股票（universe）

    #数据获取（策略需要数据）：市盈率数据（PE），市净率（PB），流动比率（CurrentRatio），长期资本（LongDebtToWorkingCapital），净利润增长率（NetProfitGrowRate）
    data = DataAPI.MktStockFactorsOneDayGet(tradeDate=previous_date,secID=universe,ticker=u""
                                ,field=u"secID,tradeDate,PE,PB,CurrentRatio,LongDebtToWorkingCapital,NetProfitGrowRate",pandas="1")

    #数据处理部分（数据结构部分）：设定索引
    data = data.dropna()                                                           #去掉无效数据
    #数据处理部分（策略计算部分）：
#   data['PE'] = 1.0 / data['PE']                                                  # PE转化为EP
    #策略构建部分：选择积极股（股票市盈率低于市场平均水平，股票的市净率低于3，企业的流动比率大于1.1，企业的长期负债与营运资金(流动资产-流动负债)比率不超过5，净利润增长率处于较高水平）生成buylist
    data = data[(data.PB < 3) & (data.PB > 0) & (data.PE > data.PE.quantile(0.8)) & (data.CurrentRatio > 1.1) & (data.LongDebtToWorkingCapital < 5)  & (data.NetProfitGrowRate >
                data.NetProfitGrowRate.quantile(0.8))]                             #通过指标选择，生成股票集合（quantile(0.8)为80%分位数）
    buylist =  data['secID'].tolist()                                              #选择积极股,生成buylist
    return  list(buylist)                                                          #返回buylist
```

积极投资法的量化投资策略的代码及注释

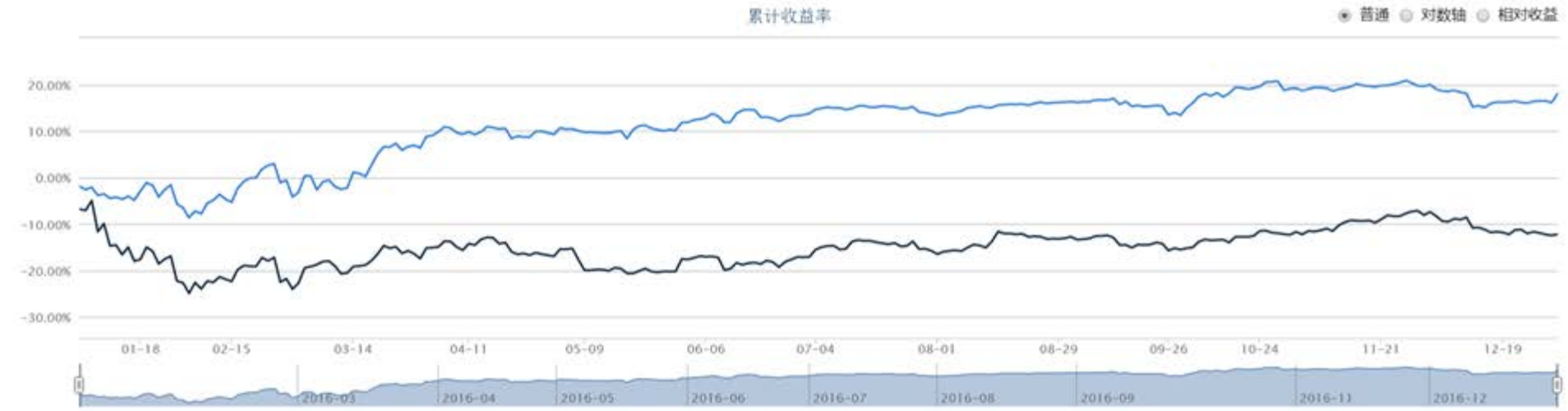

```
def trading(buylist,context):

 """
 仓位管理（固定金额）：单只股票的买入限额为1000，000元
 """

 #数据获取（通用部分）：投资者账户，可供投资股票
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

 #交易执行部分：卖出
 for sec in current_universe:
    if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 d = min(len(buylist), int(cash) // Max_Position)                                    #可以买入的股票数量，如果资金不够，只买入部分
 for sec in buylist[:d]:                                                             #买入buylist中前d只股票，d<=len(buylist)
    if sec not in security_position:
        order(sec, Max_Position / context.current_price(sec))                        #基于仓位管理的需要，每只股票最大投资额为100000元
```

成长股投资者法的量化投资策略的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
18.5%	-12.6%	23.6%	0.52	1.01	15.6%	1.93	8.7%	2.13

回测详情　开始交易



专业来自101%的投入!

模块复用：选择满足条件的因子

金程教育 GOLDEN FUTURE

```
#3.模块复用：选择满足条件的因子
  #数据获取（策略需要数据）：市盈率数据（PE），市净率（PB），流动比率（CurrentRatio），长期资本（LongDebtToWorkingCapital），净利润增长率（NetProfitGrowRate）
data = DataAPI.MktStockFactorsOneDayGet(tradeDate=previous_date,secID=universe,ticker=u""
                                        ,field=u"secID,tradeDate,PE,PB,CurrentRatio,LongDebtToWorkingCapital,NetProfitGrowRate",pandas="1")
    #数据处理部分（数据结构部分）： 设定索引
data = data.dropna()                                                                    #去掉无效数据
    #数据处理部分（策略计算部分）：
  #  data['PE'] = 1.0 / data['PE']                                                      # PE转化为EP
    #策略构建部分：选择积极股（股票市盈率低于市场平均水平，股票的市净率低于3，企业的流动比率大于1.1，企业的长期负债与营运资金(流动资产-流动负债)比率不超过5，净利润增长率处于较高水平）生成buylist
data = data[(data.PB < 3) & (data.PB > 0) & (data.PE > data.PE.quantile(0.8)) & (data.CurrentRatio > 1.1) & (data.LongDebtToWorkingCapital < 5)  & (data.NetProfitGrowRate >
            data.NetProfitGrowRate.quantile(0.8))]                                      #通过指标选择，生成股票集合（quantile(0.8)为80%分位数）
```

史蒂夫·路佛的价值投资策略

- 史蒂夫·路佛简介
- 价值投资策略的基本原理及操盘策略
- 价值投资策略的应用举例
- 价值投资策略的量化投资策略的代码及注释
- 价值投资策略的量化投资策略的回测结果
- 模块复用：指定最高投资金额的交易策略

史蒂夫·路佛的简介

- 史蒂夫.路佛（1938—）从事投资研究40年，其独特的见解经常被刊登在美国著名的研究期刊中，并受到华尔街专业投资者的一致推崇。
  - 1969-1977年间担任Piper, Jaffray Hopwood公司的投资策略分析师。
  - 1977-1981年间担任德州Criterion Investment Management公司的投资组合经理。
  - 1980年出版的《The Myths of inflation and Investing》堪称投资史上的经典。
  - 1981年创建路佛集团，一个国际化的投资研究公司。1987年开始，路佛集团集团开始利用内部的研究成果开展证券投资业务，正式成为资产管理公司。
  - 2001年3月22日，路佛核心投资基金以5年、3年、1年及3个月报酬率皆打败S&P500指数的优异表现，获得晨星公司(MorningstarInc.)美国本土混合型基金类(DomesticHybrid)五颗星的评价。
  - 2005年，路佛公司管理的资产规模已经达到10亿美元。

价值投资策略的基本原理及操作步骤

- **基本原理：**
  - 史蒂夫·路佛价值选股法则如下：
    - ✓ 市净率低于全市场平均值的80%且小于1.5倍
    - ✓ 以五年平均盈余计算的PE低于全市场平均值的70%且小于12倍
    - ✓ 每股现金至少是股价的15%
    - ✓ 股息收益率不低于全市场平均值且不低于3%
    - ✓ 股价现金流量比低于全市场平均值的75%
    - ✓ 长期借款加未提拨退休金负债占总资本比率低于50%
    - ✓ 流动比率高于全市场平均值
  - 结合中国实际情况后，原始标准调整如下：
    - ✓ 市净率低于全市场均值的80%
    - ✓ 以五年平均盈余计算的PE低于全市场平均值的70%
    - ✓ 不考虑股息收益率因素（中国股利的派发情况比较稀少）
    - ✓ 股价现金流量比低于全市场平均值的75%
    - ✓ 长期借款占总资本比率低于50%
    - ✓ 流动比率高于全市场均值
- **操作策略：**
  - 买入条件：满足中国市场6条规则，则买入
  - 卖出条件：不满足中国市场6条规则，则卖出

Current ratio = Current assets / Current liabilities

Quick ratio

= [cash + marketable securities + receivable] / Current liabilities

= [current asset - inventories] / Current liabilities

Cash ratio

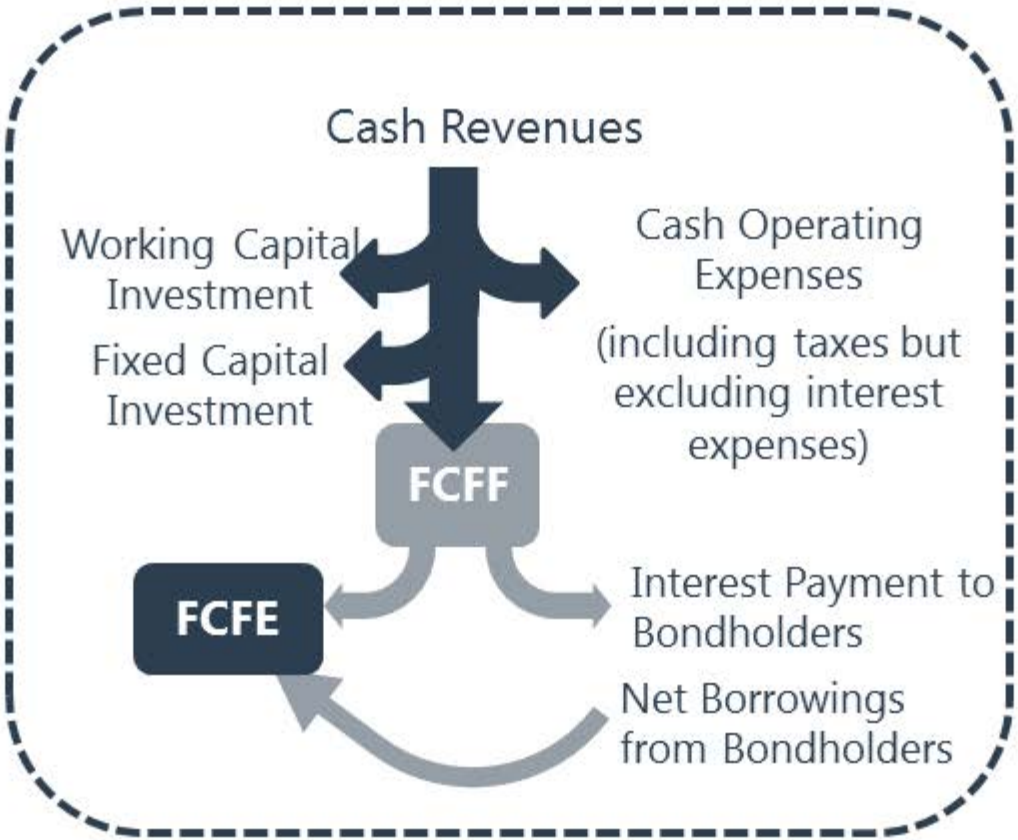
= [cash + marketable securities] / Current liabilities

➢**Advantages**

- Cash flow less subject to manipulate than EPS.
- More stable than P/E
- Handles the problem of differences in the quality of reported earning
- Empirical evidence supported

➢**Disadvantages**

- Difficult to estimate true CFO
- FCFE better but more volatile and more frequently negative.


Cash Revenues
Working Capital Investment
Fixed Capital Investment
Cash Operating Expenses
(including taxes but excluding interest expenses)
FCFF
FCFE
Interest Payment to Bondholders
Net Borrowings from Bondholders

价值投资策略的应用举例

金程教育 GOLDEN FUTURE

secID	tradeDate	PB	ETP5	PCF	CurrentRatio	LongDebtToAsset
000088.XSHE	2016/2/29	2.1087	0.0422	31.9611	5.0591	0.1543
000088.XSHE	2016/4/26	2.3833	0.037	823.94	3.2269	0.1496

101%的投入!

价值投资法的量化投资策略的代码及注释


```
#加载库函数
import datetime                          #加载时间函数库
import numpy as np
import pandas as pd
#初始化回测环境
start = '20160301'                       # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20160901'                         # 回测结束时间
benchmark = '000002.ZICN'                # 策略参考标准为A股指数
universe = set_universe('A')             # 证券池：可供选择的股票的范围为A股.（由于选股条件比较苛刻，所以为了保证有充足的可供选择股票，因此以整个A股作为投资范围）
freq = 'd'                               # 用日线回测的策略
refresh_rate = 10                        # 每10日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position = 1000000                   #每只股票的买入限额为1,000,000元

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    buylist = stock_sellection_SteveLouf_value_stock(context)  # 基于史蒂夫·路佛价值投资思想的选股策略
    trading(buylist,context)                                   # 基于固定投资额的仓位管理策略
```

价值投资法的量化投资策略的代码及注释

```
def stock_sellection_SteveLouf_value_stock(context):

    """
    操盘规则(预测)：满足以下条件买入（如已买入则持有），否则卖出
  1. 市净率低于全市场均值的80%
  2. 以五年平均盈余计算的PE低于全市场平均值的70%
  3. 不考虑股息收益率因素（中国股利的派发情况比较稀少）
  4. 股价现金流量比低于全市场平均值的75%
  5. 长期借款占总资本比率低于50%
  6. 流动比率高于全市场均值
    """

    #数据获取（通用部分）：投资者账户，可供投资股票
    previous_date = context.previous_date                                          #获取上一交易日的时间
    previous_date = previous_date.strftime('%Y%m%d')                               #将日期格式调整为%Y%m%d形式（20150227），方便读取因子数据
    account = context.get_account('fantasy_account')                               #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)  #获取当前除停牌外的所有可供投资股票（universe）

    #数据获取（策略需要数据）：SV_factor(SteveLouf的价值因子)市净率（PB），流动比率（CurrentRatio），长期借款占总资本比率（LongDebtToAsset），5年平均收益市值比（ETP5），股价现金流量比（PCF）
    SV_factor = DataAPI.MktStockFactorsOneDayGet(tradeDate=previous_date,secID=current_universe,
                                                 field=['secID','PB','ETP5','PCF','CurrentRatio','LongDebtToAsset'],pandas="1")
    #数据处理部分（数据结构部分）： 去掉无效数据并设定索引
    SV_factor = SV_factor.dropna()                                                 #去掉无效数据
    SV_factor = SV_factor.set_index(['secID'])                                     #设置索引
    #数据处理部分（逻辑部分）：去掉现金流为负的股票
    SV_factor = SV_factor[SV_factor['PCF'] > 0]                                    #去掉现金流为负的股票

    #策略构建部分：选择史蒂夫•路佛价值股（市净率低于全市场均值的80%，以五年平均盈余计算的PE低于全市场平均值的70%，股价现金流量比低于全市场平均值的75%，长期借款占总资本比率低于50%，流动比率高于全市场均值）
                 生成buylist
    SV_factor = SV_factor[(SV_factor['ETP5'] > 0.7*SV_factor['ETP5'].mean()) & (SV_factor['PCF'] < 0.75*SV_factor['PCF'].mean()) & (SV_factor['LongDebtToAsset'] < 0.5) &
                          (SV_factor['PB'] < 0.8*SV_factor['PB'].mean()) & (SV_factor['CurrentRatio']> SV_factor['CurrentRatio'].mean())]
    buylist =  SV_factor.index                                                     #选择SteveLouf价值股，生成buylist
    return  list(buylist)                                                          #返回buylist
```

价值投资法的量化投资策略的代码及注释

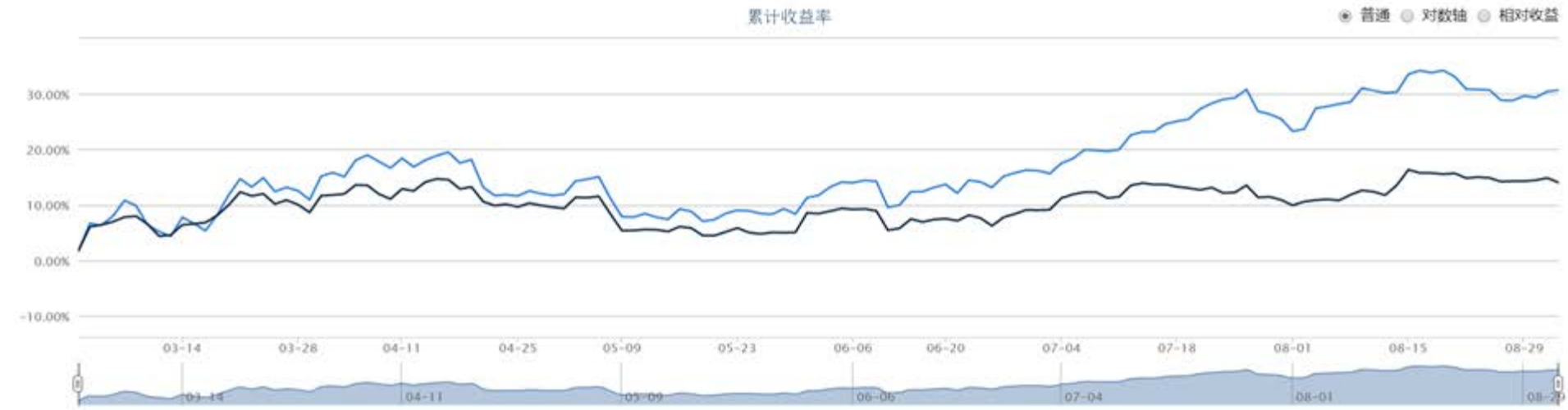

```
def trading(buylist,context):

 """
 仓位管理（固定金额）：单只股票的买入限额为1000，000元
 """

 #数据获取（通用部分）：投资者账户，可供投资股票
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

 #交易执行部分：卖出
 for sec in current_universe:
    if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 d = min(len(buylist), int(cash) // Max_Position)                                    #可以买入的股票数量，如果资金不够，只买入部分
 for sec in buylist[:d]:                                                             #买入buylist中前d只股票，d<=len(buylist)
    if sec not in security_position:
        order(sec, Max_Position / context.current_price(sec))                        #基于仓位管理的需要，每只股票最大投资额为100000元
```

价值投资法的量化投资策略的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
68.0%	28.9%	34.0%	1.18	2.90	23.4%	2.31	10.4%	0.78

回测详情　开始交易



专业来自101%的投入!

模块复用：指定最高投资金额的交易策略


```
def trading(buylist,context):

 """
 仓位管理（固定金额）：单只股票的买入限额为1000，000元
 """

 #数据获取（通用部分）：投资者账户，可供投资股票
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

 #交易执行部分：卖出
 for sec in current_universe:
    if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 d = min(len(buylist), int(cash) // Max_Position)                                    #可以买入的股票数量，如果资金不够，只买入部分
 for sec in buylist[:d]:                                                             #买入buylist中前d只股票，d<=len(buylist)
    if sec not in security_position:
        order(sec, Max_Position / context.current_price(sec))                        #基于仓位管理的需要，每只股票最大投资额为100000元
```

惠特尼•乔治小型价值股投资策略

- 惠特尼•乔治简介
- 小型价值股投资策略的基本原理及操盘策略
- 小型价值股投资策略的应用举例
- 小型价值股投资策略的量化投资策略的代码及注释
- 小型价值股投资策略的量化投资策略的回测结果
- 模块复用：指定最高投资比例的交易策略

惠特尼•乔治的简介

- 惠特尼•乔治出生于俄亥俄州的代顿市（Dayton），投资经历长达30年以上。乔治(Whitney George)和他的团队自称市场的“清道夫”，擅长投资小企业，钟情于投资股价低于25美元的小盘股票。
- 惠特尼•乔治认为股票中有很大一部分被投资者所忽视。乔治表示他团队的目标就是买入那些在未来三五年内股价能够翻番甚至更多的股票。尽管这些股票有可能由于经营不及预期或行业不被看好而股价大跌，但那些“具有稳定的资产负债表和长期健康股本回报的公司，特别是能够通过扩展现存业务或者进行新的收购而在未来几年实现增长的公司”仍然值得投资。他坚信小股本企业股票会在低收益的环境下表现更好，因为小企业通常比大企业具有更高的灵活性。
- 乔治在2011年1月被《财智月刊》评为美国小盘股基金“最佳基金经理”。

小型价值股投资策略的基本原理及操作步骤

- **基本原理：**
  - 市净率低于全市场平均值 （PB）
  - 市盈率小于市场平均值；(PE(TTM))
  - 市销率小于市场平均值 (PS(TTM))
  - 总市值小于全市场中位数市值
  - 现金流市值比(红利)大于市场平均值 （CTOP）
  - 产权比率（负债总额与所有者权益总额的比率）小于全市场平均值 （DtE）
  - 权益回报率大于市场平均值 (ROE)
  - 资产回报率大于市场平均值 (ROA)
- **操作策略：**
  - 买入条件： 满足惠特尼•乔治8项指标条件的股票,买入
  - 卖出条件： 不满足惠特尼•乔治8项指标条件的股票,卖出

➢**Advantages:**

- Meaningful even for distressed firms.
- Sales revenue not easily manipulated.
- Not as volatile as P/E ratios.
- Useful in valuing mature, cyclical, and start-up firms.
- Differences in P/Ss may be related to differences in long-run average returns.

➢**Disadvantages:**

- High sales do not imply high profits and cash flows.
- Does not capture cost structure differences.
- Revenue recognition practices still distort sales.

Return on assets (ROA) = [NI + int.(1-t)] / average total assets

Operating return on assets = EBIT / average total assets

[Return on total capital (ROTC) = EBIT / average total capital]

Return on equity (ROE) = NI / average total equity

Return on common equity = (NI- Preferred Dividend) / average common equity

小型价值股投资策略的应用举例

金程教育 GOLDEN FUTURE

secID	PB	PE	PS	LCAP	CTOP	DebtEquityRatio	ROE	ROA
600313.XSHG	4.5255	45.1939	1.2438	22.343	0.0073	0.5968	0.1001	0.0726
600313.XSHG	4.6884	46.8204	1.2885	22.3783	0.007	0.5968	0.1001	0.0726

自10%的投入!

小型价值股投资法的量化投资策略的代码及注释


```
#加载库函数
import datetime                          #加载时间函数库
import numpy as np
import pandas as pd
#初始化回测环境
start = '20160301'                       # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20160901'                         # 回测结束时间
benchmark = '000002.ZICN'                # 策略参考标准为A股指数
universe = set_universe('A')             # 证券池：可供选择的股票的范围为A股.（由于选股条件比较苛刻，所以为了保证有充足的可供选择股票，因此以整个A股作为投资范围）
freq = 'd'                               # 用日线回测的策略
refresh_rate = 10                        # 每10日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position_per = 0.1                   #每只股票购入的最高比例为10%

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    buylist = stock_sellection_George_value_stock(context)      # 基于惠特尼•乔治价值投资思想的选股策略
    trading(buylist,context)                                    # 基于固定投资比例的仓位管理策略
```

小型价值股投资法的量化投资策略的代码及注释


```
def stock_sellection_George_value_stock(context):

    """
    操盘规则(预测)：满足以下条件买入（如已买入则持有），否则卖出
        1.市净率低于全市场平均值 （PB）
        2.市盈率小于市场平均值；(PE)
        3.市销率小于市场平均值 (PS)
        4.总市值小于全市场中位数市值
        5.现金流市值比(红利)大于市场平均值 （CTOP）
        6.产权比率（负债总额与所有者权益总额的比率）小于全市场平均值 (DebtEquityRatio)
        7.年权益回报率大于市场平均值 (ROE)
        8.年资产回报率大于市场平均值 (ROA)
    """

    #数据获取（通用部分）：投资者账户，可供投资股票
    previous_date = context.previous_date                                          #获取上一交易日的时间
    previous_date = previous_date.strftime('%Y%m%d')                               #将日期格式调整为%Y%m%d形式（20150227），方便读取因子数据
    account = context.get_account('fantasy_account')                              #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)  #获取当前除停牌外的所有可供投资股票（universe）

    #数据获取（策略需要数据）：GSV_factor(George的价值因子)：市净率（PB），市盈率（PE），市销率（PS），对数市值（LCAP），
                                现金流市值比（CTOP），财务杠杆（DebtEquityRatio），资产回报率（ROA），权益回报率(ROE)
    GSV_factor = DataAPI.MktStockFactorsOneDayGet(tradeDate=previous_date,secID=current_universe,
                                    field=['secID','PB','PE','PS','LCAP','CTOP','DebtEquityRatio','ROE','ROA'],pandas="1")
    #数据处理部分（数据结构部分）： 去掉无效数据并设定索引
    GSV_factor = GSV_factor.dropna()                                               #去掉无效数据
    GSV_factor = GSV_factor.set_index(['secID'])                                   #设置索引
    #数据处理部分（逻辑部分）：去掉现金流为负的股票
    GSV_factor = GSV_factor[GSV_factor['PE'] > 0]                                  #去掉现金流为负的股票

    #策略构建部分：George的小型价值股（市净率低于全市场平均值 （PB），市盈率小于市场平均值，市销率小于市场平均值，总市值小于全市场中位数市值，现金流市值比(红利)大于市场平均值 （CTOP），
                产权比率（负债总额与所有者权益总额的比率）小于全市场平均值 (DebtEquityRatio)，权益回报率大于市场平均值 (ROE5)，资产回报率大于市场平均值 (ROA5)）生成buylist
    GSV_factor = GSV_factor[(GSV_factor['PB'] < GSV_factor['PB'].mean()) & (GSV_factor['PE'] < GSV_factor['PE'].mean()) & (GSV_factor['PS'] < GSV_factor['PS'].mean()) &
                        (GSV_factor['LCAP'] < GSV_factor['LCAP'].median()) & (GSV_factor['DebtEquityRatio'] < GSV_factor['DebtEquityRatio'].mean()) &
                        (GSV_factor['ROE'] > GSV_factor['ROE'].mean()) & (GSV_factor['ROA'] > GSV_factor['ROA'].mean())]
    buylist =  GSV_factor.index                                                    #选择SteveLouf价值股,生成buylist
    return  list(buylist)                                                          #返回buylist
```

小型价值股投资法的量化投资策略的代码及注释

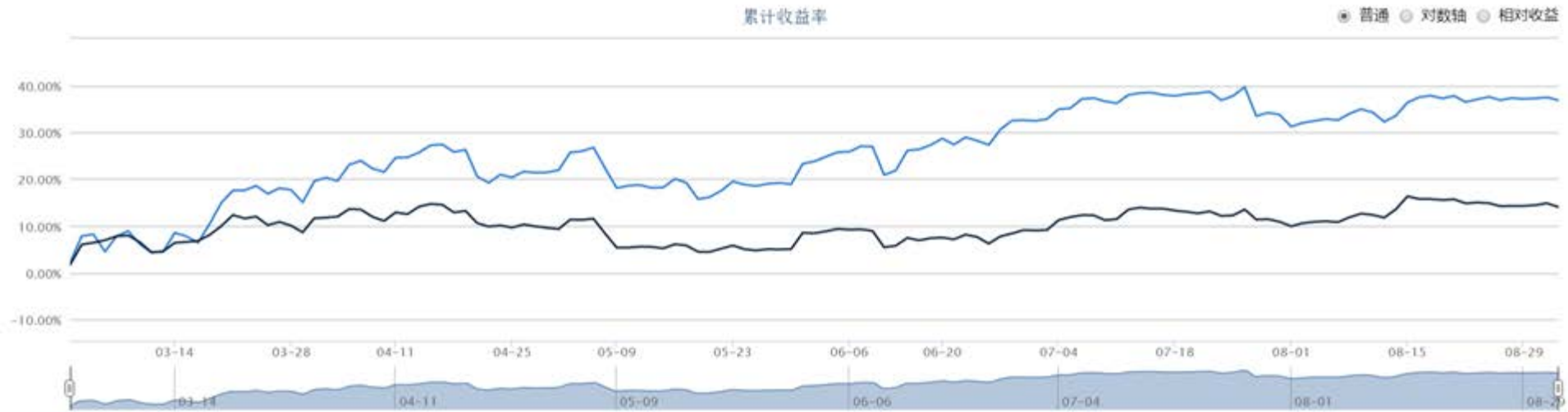

```
def trading(buylist,context):

 """
 仓位管理：单只股票的买入限额为全部资金的10%
 """

 #数据获取（通用部分）：投资者账户，可供投资股票
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

 #交易执行部分：卖出
 for sec in current_universe:
    if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 if len(buylist) > 0:                                                                #判读本期是否有需要买入股票（buylist>0）
            weight = min(Max_Position_per,1.0/len(buylist))                          #每只股票所分配的金额相同，但最多为总金额的10%
 else:
     weight = 0                                                                      #若本期没有要购入股票，设置分配权重为0
 for sec in buylist:                                                                 #判断股票是否为本期要购入股票且尚未持有
     if sec not in security_position:
        order_pct_to(sec,weight)                                                     #购入指定比例的股票
```

价值投资法的量化投资策略的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
83.6%	28.9%	45.1%	1.33	3.20	26.1%	2.74	9.2%	1.43

回测详情　开始交易



专业来自101%的投入!

模块复用：指定最高投资比例的交易策略

金程教育 GOLDEN FUTURE

```
def trading(buylist,context):

    """
    仓位管理：单只股票的买入限额为全部资金的10%
    """

    #数据获取（通用部分）：投资者账户，可供投资股票
    account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
    security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
    cash = account.cash                                                                 #获取股票账户可用于投资的现金额度

    #交易执行部分：卖出
    for sec in current_universe:
       if sec not in buylist and sec in security_position:                              #不在buylist中，且有持仓，卖出
           order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
    #交易执行部分：买入
    if len(buylist) > 0:                                                                #判读本期是否有需要买入股票（buylist>0）
            weight = min(Max_Position_per,1.0/len(buylist))                             #每只股票所分配的金额相同，但最多为总金额的10%
    else:
        weight = 0                                                                      #若本期没有要购入股票，设置分配权重为0
    for sec in buylist:                                                                 #判断股票是否为本期要购入股票且尚未持有
        if sec not in security_position:
           order_pct_to(sec,weight)                                                     #购入指定比例的股票
```

CONTENTS

PROFESSIONAL · LEADING · VALUE-CREATING

专业来自101%的投入!

- 均线排列交易系统
- 金肯纳特交易系统
- 海龟交易系统

均线排列交易系统

- 均线排列交易系统的基本原理及操盘策略
- 均线排列交易系统的应用举例
- 均线排列交易系统的代码及注释
- 均线排列交易系统的回测结果
- 模块复用：指定买卖列表的交易策略

基本原理：

- 股价排列表现的是股价变动的中期趋势，代表了股价一定期限内的运行方向。
  - ✓ 价多头排列是指股价的短期5日均线、10日均线上方运行，下面的20日均线在下方依次顺序排列成多头排列。这意味着中短期的上涨趋势已经形成，并且不同均线成本位置的人达成了一致的看法，所以股价向上运行，股价逐渐抬高，方向向上；
  - ✓ 股价空头排列是指最上方短期5日均线的投资者获得一定利润的时候，就会抛出股票，产生抛压后股价会跌落到10日均线，然后依此类推抛压产生，均线逐级跌破，均线的排列最后呈现从大到小。最后呈现空头排列，投资者们达成了共识一致做空，所以市场上经常说做多头排列的股票，不做空头排列的股票。
- 损失厌恶反映了这样一种非理性行为：当涉及的是收益时，人们表现为风险厌恶；当涉及的是损失时，人们则表现为风险寻求。因此在短期投资中可用来确定因投资者情绪导致的短期的买卖点。
- 因此可以在中期趋势满足的前提下，使用短期信号提示买卖点

操作策略：

- 入场条件 均线（5日，10日，20日）多头排列，且当前价格小于或等于5日均线
- 出场条件 5日均线下穿10日均线，或当前价格小于10日均线，或价格严重脱离均线（5%）

Loss aversion. Investors are more risk averse when faced with potential losses and less risk averse when faced with potential gains.

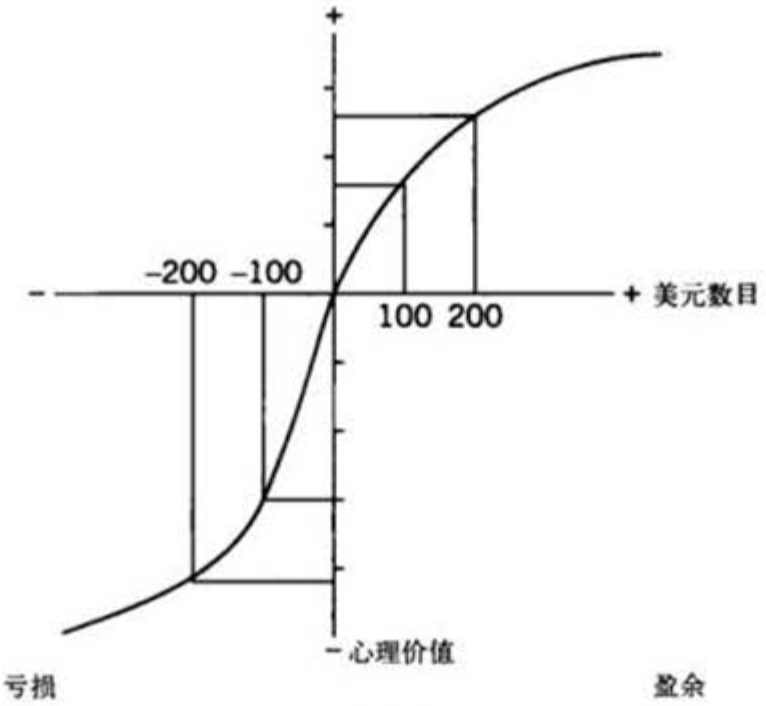


图 10

均线排列交易系统的应用举例

均线排列交易系统的代码及注释


```
#加载库函数
import numpy as np
import pandas as pd
import talib as ta                      # 加载技术分析库

#初始化回测环境
start = '20141201'                      # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20150601'                        # 回测结束时间
benchmark = 'HS300'                     # 策略参考标准为A股指数
universe = DynamicUniverse('HS300').apply_filter(Factor.PE.nlarge(100))   # 证券池：可供选择的股票的范围为A股，（由于选股条件比较苛刻，所以为了保证有充足的可供选择股票，因此以整个A股作为投资范围）
freq = 'd'                              # 用日线回测的策略
refresh_rate = 1                        # 每1d日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position_per = 0.1                  # 每只股票购入的最高比例为10%
max_history_window = 250                # 设置最长回测周期
Max_time_range = 60                     # 构建策略最长回看时间
fastLine = 5                            # 短周期快线周期参数
midLine  = 10                           # 中等周期周期参数
slowLine = 20                           # 长周期慢线周期参数
threshold = 0.05                        # 止盈收益率

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    tradingDic = timing_MA(context)     # 基于均线排列的择时策略
    trading(tradingDic,context)         # 基于固定投资比例的仓位管理策略
```

均线排列交易系统的代码及注释

金程教育 GOLDEN FUTURE

```
def timing_MA(context):
    ##均线（5日，10日，20日）多头排列，且当前价格小于或等于5日均线
    ##5日均线下穿10日均线，或当前价格小于10日均线

    #数据获取（通用部分）：投资者账户，可供投资股票，价格数据，持仓数据，账户金额数据
account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
history = context.history(current_universe,['closePrice','lowPrice'],Max_time_range, rtype='array')   #拿过去Max_time_range个交易日的收盘价来构建均线排列系统
security_position = account.get_positions()                                          #字典数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸

    #择时策略部分：获取当前时点的buylist和selllist
buylist = []                                                                         #初始化购买股票列表
selllist = []                                                                        #初始化卖出股票列表
for sec in current_universe:                                                         #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票
  close = history[sec]['closePrice']                                                 #获取股票sec过去Max_time_range天的收盘价
  low   = history[sec]['lowPrice']                                                   #获取股票sec过去Max_time_range天的最低价
  fast = ta.MA(close, fastLine)                                                      #获得短期均线
  mid = ta.MA(close, midLine)                                                        #获得中期均线
  long = ta.MA(close, slowLine)                                                      #获得长期均线
  if  fast[-1] > mid[-1] and mid[-1] > long[-1] and close[-1] <= fast[-1] and sec not in security_position: #股票均线多头排列,股价小于5
                                                                                                          日均线，且未持有该股票
      buylist.append(sec)
  elif ((fast[-2] > mid[-2] and fast[-1] < mid[-1]) or (close[-1]<= mid[-1]) or (low[-1]- fast[-1])/fast[-1] > threshold) and sec in security_position:  #短期均线下穿长期均线或收盘价跌
                                                                                                                                                    破10日均线，且已持有该股票
      selllist.append(sec)                                                           #取票加入selllist

    #返回交易字典
TradingDic={'buyList':buylist,'sellList':selllist}                                   #TradingDic 为字典型变量，以listName为key，list为Value
return TradingDic                                                                    #返回交易字典
```


专业来自101%的投入！

均线排列交易系统的代码及注释

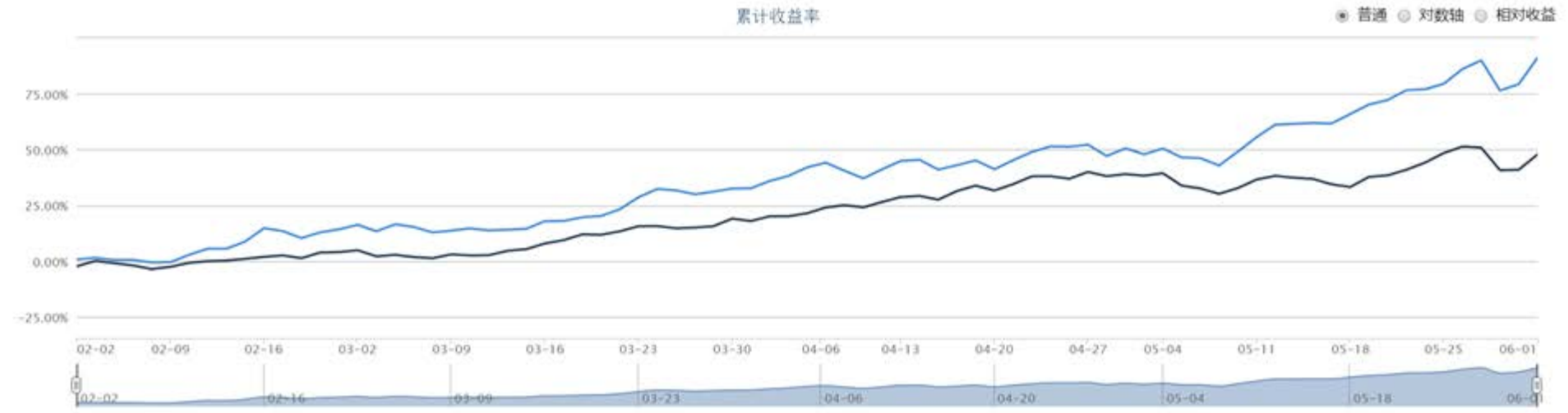

```
def trading(tradingDic,context):

 """
 仓位管理：单只股票的买入限额为全部资金的10%
 """

 #数据获取（通用部分）：投资者账户，可供交易的股票，投资者持仓及交易数据
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度
 buylist = tradingDic.get('buyList')                                                 #获取购入股票数据
 selllist = tradingDic.get('sellList')                                               #获得卖出股票数据

 #交易执行部分：卖出
 for sec in current_universe:
    if sec in selllist and sec in security_position:                                 #在selllist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 if len(buylist) > 0:                                                                #判读本期是否有需要买入股票（buylist>0）
        weight = min(Max_Position_per,1.0/len(buylist))                              #每只股票所分配的金额相同，但最多为总金额的10%
 else:
     weight = 0                                                                      #若本期没有要购入股票，设置分配权重为0
 for sec in buylist:                                                                 #判断股票是否为本期要购入股票且尚未持有
     if sec not in security_position:
        order_pct_to(sec,weight)                                                     #购入指定比例的股票
```

均线排列交易系统的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
677.1%	244.3%	448.8%	0.93	18.74	36.0%	3.46	7.1%	14.12

回测详情　开始交易



专业来自101%的投入!

模块复用：指定买卖列表的交易策略

金程教育 GOLDEN FUTURE

```
def trading(tradingDic,context):

 """
 仓位管理：单只股票的买入限额为全部资金的10%
 """

 #数据获取（通用部分）：投资者账户，可供交易的股票，投资者持仓及交易数据
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度
 buylist = tradingDic.get('buyList')                                                 #获取购入股票数据
 selllist = tradingDic.get('sellList')                                               #获得卖出股票数据

 #交易执行部分：卖出
 for sec in current_universe:
    if sec in selllist and sec in security_position:                                 #在selllist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 if len(buylist) > 0:                                                                #判读本期是否有需要买入股票（buylist>0）
        weight = min(Max_Position_per,1.0/len(buylist))                              #每只股票所分配的金额相同，但最多为总金额的10%
 else:
     weight = 0                                                                      #若本期没有要购入股票，设置分配权重为0
 for sec in buylist:                                                                 #判断股票是否为本期要购入股票且尚未持有
     if sec not in security_position:
        order_pct_to(sec,weight)                                                     #购入指定比例的股票
```

金肯纳特交易系统

- 金肯纳特交易系统的基本原理及操盘策略
- 金肯纳特交易系统的应用举例
- 金肯纳特交易系统的代码及注释
- 金肯纳特交易系统的回测结果
- 模块复用：期货交易系统向股票交易系统的迁移

基本原理：

- 传统通道突破策略是建立在通道突破思想上的，突破上轨做多；下破下轨做空。但是通道突破系统主要的问题在于假突破。假突破中，通道的突破并没有带来趋势的确定，反而是价格动能的衰竭，会迅速回落并反向运动。
- 金肯特纳交易系统针对假突破问题进行了改进。金肯特纳交易系统在移动平均线附近设置了一个止损，以限制假突破造成的损失。具体而言，三价（最高价、最低价和收盘价）均线向上，并且价格上破通道上轨，买入股票，价格下破三价均线，卖出股票。其中上轨=三价均线+真实振幅的移动平均值

操作策略：

- 入场条件：三价均线向上，并且价格上破通道上轨，买入股票
- 出场条件：价格下破三价均线，卖出股票

基本原理

- 主要应用于了解股价的震荡幅度和节奏，在窄幅整理行情中用于寻找突破时机。通常情况下股价的波动幅度会保持在一定常态下，但是如果有主力资金进出时，股价波幅往往会加剧，甚至突破移动平均值和真实振幅所构成的通道，由此提示买卖时机。

计算公式

$$TR = \boldsymbol{max}\{|\text{最高价} - \text{最低价}|, |\text{最高价} - \text{收盘价}|, |\text{收盘价} - \text{最低价}|\}$$

$$\text{真实波幅}(ATR) = \text{MA}(\text{TR}, \boldsymbol{N})$$

金肯纳特交易系统的代码及注释


```
#加载库函数
import numpy as np
import pandas as pd
import talib as ta                          # 加载技术分析库

#初始化回测环境
start = '20150201'                          # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20150601'                            # 回测结束时间
benchmark = 'HS300'                         # 策略参考标准为A股指数
universe = DynamicUniverse('HS300').apply_filter(Factor.PE.nlarge(100))    # 证券池：可供选择的股票的范围为A股，（由于选股条件比较苛刻，所以为了保证有充足的可供选择股票，因此以整个A股作为投资范围）
freq = 'd'                                  # 用日线回测的策略
refresh_rate = 1                            # 每10日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position_per = 0.1                      # 每只股票购入的最高比例为10%
max_history_window = 250                    # 设置最长回测周期
Max_time_range = 60                         # 构建策略最长回看时间
atrlength  = 20                             # 计算真实波幅所需时间周期
avglength = 20                              # 计算三价均值所需时间周期


#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    tradingDic = timing_CK(context)    # 金肯特纳交易系统
    trading(tradingDic,context)        # 基于固定投资比例的仓位管理策略
```

金肯纳特交易系统的代码及注释


```
def timing_CK(context):
    """
    三价均线向上，并且价格向上突破上轨，开多单；持有多单时，并且价格向下突破三价均线，平多单
    """
    #数据获取（通用部分）：投资者账户，可供投资股票，价格数据，持仓数据，账户金额数据
    account = context.get_account('fantasy_account')                                                                    #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)                                     #获取当前除停牌外的所有可供投资股票（universe）
    history = context.history(current_universe,['closePrice','lowPrice','highPrice'],Max_time_range, rtype='array')     #构建金肯特纳交易系统时考虑过去Max_time_range的历史数据
    security_position = account.get_positions()                                                                          #字典数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸

    #按照金肯特纳交易系统循环处理所有证券池中股票
    buylist = []                                                                                                         #初始化购买股票列表
    selllist = []                                                                                                        #初始化卖出股票列表
    for sec in current_universe:                                                                                         #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票

    #数据获取（专有部分）：最低价，最高价，收盘价
        close = history[sec]['closePrice']                                                                                 #获取股票sec过去Max_time_range天的收盘价
        low   = history[sec]['lowPrice']                                                                                   #获取股票sec过去Max_time_range天的最低价
        high  = history[sec]['highPrice']                                                                                  #获取股票sec过去Max_time_range天的最高价

    #数据处理（策略部分）：计算真实波幅，通道上轨以及离场点
        atr = ta.ATR(high, low, close, atrlength)[-1]                                                                      #计算真实波幅
        movavgval = ta.MA((high+low+close)/3, avglength)                                                                   #计算最高、最低和收盘价三价均线
        upband = movavgval[-1] + atr                                                                                       #基于真实波幅的通道上轨
        liquidpoint = movavgval                                                                                            #离场点为三价均值
    #策略部分：三价均线向上，并且价格向上突破上轨，开多单；持有多单时，并且价格向下突破三价均线，平多单
        if  movavgval[-1] > movavgval[-2] and high[-1] >= upband and sec not in security_position:                          #三价均线向上，并且价格向上突破上轨，开多单
            buylist.append(sec)                                                                                            #将股票加入buylist
        elif low[-1] <= liquidpoint[-1] and sec in security_position:                                                      #持有多单时，并且价格向下突破三价均线，平多单
            selllist.append(sec)                                                                                           #将股票加入selllist

    #返回交易字典
    TradingDic={'buyList':buylist,'sellList':selllist}                                                                   #TradingDic 为字典型变量，以listName为key，list为Value
    return TradingDic                                                                                                    #返回交易字典
```

金肯纳特交易系统的代码及注释


```
def trading(tradingDic,context):

 """
 仓位管理：单只股票的买入限额为全部资金的10%
 """

 #数据获取（通用部分）：投资者账户，可供交易的股票，投资者持仓及交易数据
 account = context.get_account('fantasy_account')                                    #获取投资者的股票账户（fantasy_account）
 current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True)     #获取当前除停牌外的所有可供投资股票（universe）
 security_position = account.get_positions()                                         #字典型数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
 cash = account.cash                                                                 #获取股票账户可用于投资的现金额度
 buylist = tradingDic.get('buyList')                                                 #获取购入股票数据
 selllist = tradingDic.get('sellList')                                               #获得卖出股票数据

 #交易执行部分：卖出
 for sec in current_universe:
    if sec in selllist and sec in security_position:                                 #在selllist中，且有持仓，卖出
        order_to(sec,0)                                                              #执行卖出指令后,会自动更新股票账户中的金额
 #交易执行部分：买入
 if len(buylist) > 0:                                                                #判读本期是否有需要买入股票（buylist>0）
        weight = min(Max_Position_per,1.0/len(buylist))                              #每只股票所分配的金额相同，但最多为总金额的10%
 else:
     weight = 0                                                                      #若本期没有要购入股票，设置分配权重为0
 for sec in buylist:                                                                 #判断股票是否为本期要购入股票且尚未持有
     if sec not in security_position:
        order_pct_to(sec,weight)                                                     #购入指定比例的股票
```

金肯纳特交易系统的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
562.2%	244.3%	362.4%	0.82	17.86	31.3%	3.07	5.6%	3.90

回测详情　开始交易

专业来自101%的投入!

模块复用：期货交易系统向股票交易系统的迁移


```
for sec in current_universe:                                                    #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票

#数据获取（专有部分）：最低价，最高价，收盘价
    close = history[sec]['closePrice']                                          #获取股票sec过去Max_time_range天的收盘价
    low   = history[sec]['lowPrice']                                            #获取股票sec过去Max_time_range天的最低价
    high  = history[sec]['highPrice']                                           #获取股票sec过去Max_time_range天的最高价

#数据处理（策略部分）：计算真实波幅，通道上轨以及离场点
    atr = ta.ATR(high, low, close, atrlength)[-1]                               #计算真实波幅
    movavgval = ta.MA((high+low+close)/3, avglength)                            #计算最高、最低和收盘价三价均线
    upband = movavgval[-1] + atr                                                #基于真实波幅的通道上轨
    liquidpoint = movavgval                                                     #离场点为三价均值
#策略部分：三价均线向上，并且价格向上突破上轨，开多单；持有多单时，并且价格向下突破三价均线，平多单
    if  movavgval[-1] > movavgval[-2] and high[-1] >= upband and sec not in security_position:  #三价均线向上，并且价格向上突破上轨，开多单
        buylist.append(sec)                                                     #将股票加入buylist
    elif low[-1] <= liquidpoint[-1] and sec in security_position:               #持有多单时，并且价格向下突破三价均线，平多单
        selllist.append(sec)                                                    #将股票加入selllist
```

海龟交易系统交易系统

- 海龟交易系统的前世今生
- 海龟交易系统的成功之道
- 海龟交易系统的操盘策略及应用举例
- 海龟交易系统的代码及注释
- 海龟交易系统的回测结果
- 模块复用：投资历程的记录、查询与更新

海龟交易系统的前世今生

- 著名的商品投机家理查德.丹尼斯与他的老友比尔.埃克哈特进行了一场辩论，这场辩论是关于伟大的交易员是天生造就还是后天培养的。理查德相信，他可以教会人们成为伟大的交易员。比尔则认为遗传和天性才是决定因素。理查德.丹尼建议招募并培训一些交易员，给他们提供真实的帐户进行交易，看看两个人中谁是正确的。
- 他们在《巴伦氏》、《华尔街期刊》和《纽约时报》上刊登了大幅广告，招聘交易学员。广告中称，在一个短暂的培训会后，新手将被提供一个帐户进行交易。因为里克（理查德的昵称）或许是当时世界上最著名的交易员，所以，有1000多位申请人前来投奔他。他会见了其中的80位。这一群人精选出10个人，后来这个名单变成13个人，所增加的3个人里克以前就认识。
- 1983年12月底，理查德.丹尼邀请被选中的13个人到芝加哥进行两周的培训，到1984年1月初，学员开始用小帐户进行交易。到了2月初，10名学员在证明了自己的能力，丹尼斯给被选中的学员提供了50到200万美元的资金帐户。

海龟交易系统的前世今生

- 在随后的4年，海龟成为交易史上最著名的实验，因为在随后的四年中学员取得了年均复利80%的收益。
- 海龟们（理查德.丹尼斯将学员们称为‘海龟’）商定在他们议定的10年保密协议于1993年终止后不泄露海龟法则，但个别海龟在网站上出售海龟交易系统获取利润。
- 柯蒂斯.菲斯（最成功的海龟之一，年化收益率超过100%）为阻止个别海龟对知识产权的偷窃和出售海龟法则而赚钱的行为，决定在网站上将海龟交易法则免费公之于众。这也是海龟交易法则的出处
- 柯蒂斯.菲斯总结了海龟交易系统本身、海龟的训练过程、海龟的交易历程以及海龟交易系统成功的本质（心理管理和仓位控制），即海龟贯彻交易系统的能力，完成了被称为有史以来最好的5本交易学著作之一《海龟交易法》。

成功的交易：纪律性

- 对于一致性赚钱的交易，使用交易系统就是最佳的方式。如果你知道自己的 系统能够长期赚钱，你就比较容易接受信号，并且在亏损期间按照系统信号进行 交易。如果你在交易中依赖自己的判断，你可能会发现恰恰应该勇敢时你却胆怯， 而恰恰应该胆怯时你却勇敢。
- 如果你拥有一个能够赢利的交易系统，而且你虔诚地跟随这个系统，那 么，你的交易将会取得赢利，而且系统会帮助你安然摆脱难免会来自于一长串亏损或者巨额赢利的内心挣扎。
- “我说过很多次，你可以把我的交易法则（系统）登在报纸上，但没有人会遵守它们。关键是统一性和纪律性。几乎每一个人都可以列出一串法则，而不比我们的那些法则（海龟交易系统）差多少。但他们不能给别人信心，而唯有对法则充满信心，你才能坚持这些法则，即使遭遇逆境。”（理查德.丹尼斯）

成功的交易系统：统一性与完整性

- 一个良好的交易系统可以自动运行整个交易程序。对于交易员在交易中 必须制定的每项决策，系统都会给出答案。该系统使交易员更容易进行一致性的 交易，因为有一套明确说明应该做什么的法则。交易的机械化就是不留给交易员自己进行判断。
- 良好的交易系统包括市场——买卖什么 ；头寸规模——买卖多少；入市——何时买卖 ；止损——何时退出亏损的头寸；离市——何时退出赢利的头寸；策略——如何买卖。
- 良好的交易系统应经过详细的回测，以明确其盈利和亏损的概率以及盈利时的收益与亏损时的损失，这有利于建立投资的信心并保证交易者对纪律的遵守。

海龟交易系统的成功原因

- 海龟交易系统有较高的概率，并且在盈利时设置较高的仓位，保证了较高的胜率。
- 海龟交易系统是一个完整的交易系统。其法则覆盖了交易的各个方面，并且 不给交易员留下一点主观想象决策的余地。它具备一个完整的交易系统的所有成分。

海龟交易系统的操盘策略及应用举例

- 市场：买卖什么
- 入市：何时买卖
- 加仓：增加单位
- 止损：何时退出亏损的头寸
- 离市：何时退出赢利的头寸
- 策略：如何买卖。

市场：买卖什么

- 交易系统的第一项决策是买卖什么，或者本质上在何种市场进行交易。如果你只在很少 的几个市场中进行交易，你就大大减少了赶上趋势的机会。同时，你不想在交易 量太少或者趋势不明郎的市场中进行交易。
- 海龟交易系统买卖的是在美国芝加哥和纽约交易所交易的具有流动性的期货，包括金融期货、商品期货以及外汇等多种产品。
- 我们将其移植到中国股票投资领域。因为海龟交易系统在买卖品种上的要求实质时要求品种具有较高的流通性，而在中国最具流动性的市场时股票市场。

头寸规模：买卖多少

- 买卖多少既影响多样化，又影响资金管理。多样化就是努力在诸多投资工具上分散风险，并且通过增加抓住成功交易的机会而增加赢利的机会。正确的多样化要求在多种不同的投资工具上进行类似的（如果不是同样的话）下注。资金管理实际上是关于通过不下注过多以致于在良好的趋势到来之前就用完自己的资金来控制风险的。
- 海龟将一个基于波动性的常数百分比用作头寸规模风险的测算标准。该测算标准使头寸的美元波动性标准化。这意味着在以美元表示的数量相同的特定交易日，特定的头寸往往会上下波动（与其他市场的头寸相比），不考虑特定市场根本的波动性。
- 我们采用与海龟系统相同的基于波动性的常数百分比用作头寸规模风险的测算标准。

入市：何时买卖

- **何时买卖的决策通常称为入市决策。自动运行的系统产生入市信号，这些信号说明了进入市场买卖的明确的价位和市场条件。**
- **原始海龟用两个相关的系统入市，这两个系统都以唐奇安的通道突破系统。**
  - 系统一：以 20 日突破（突破20日最高价）为基础的偏短线系统
  - 系统二：以 50 日突破（突破50日最高价）为基础的较简单的长线系统
- **我们在策略中选择唐奇安通道的20日突破为入场条件。**
- **后续的海龟策略变种至少有6种，包括ATR通道系统，布林带通道系统，唐奇安通道系统，定时退出唐奇安通道系统，双重移动均线系统以及三重均线移动系统**

入市：头寸管理

- 波动性N值的计算
- 第一次入市时（建仓）的仓位计算

波动性N值的计算

- **N 就是 TR（True Range，实际范围）的 20 日指数移动平均，现在更普遍 地称之为 ATR。从概念上来看，N 表示单个交易日某个特定市场所造成的价格波动的平均范围，它说明了开盘价的缺口。**

$$TR = \boldsymbol{max}\{|\text{最高价} - \text{最低价}|, |\text{最高价} - \text{收盘价}|, |\text{收盘价} - \text{最低价}|\}$$

$$\text{真实波幅}(ATR) = \text{MA}(\text{TR}, \boldsymbol{N})$$

第一次入市时（建仓）的仓位计算

股票的仓位计算

- 首次建仓的时候，当捕捉到趋势，即价格突破唐奇安上轨时，买入1个unit。
- 其意义就是，让一个N值的波动与你总资金1%的波动对应，如果买入1unit单位的资产，当天震幅使得总资产的变化不超过1%

$$Unit = \frac{1\% \times Account}{\mathrm{N}}$$

- 例如，现在你有100万元资金，1%波动就是10000元。假如股票的N值为1元，1unit为10000元÷1元=10000股。也就是说，你的第一笔仓位应该是在其突破上轨（假设为10元）时立刻买入10000股，耗资10万元。

加仓：增加单位

- 通常情况下,加仓的前提是已经预测出资产价格的波动趋势,并且后期市场还会有较强的趋势的发展.只有在这个前提之下,投资人选择加仓的风险才是最低的.其实总结来说就是,头寸出现亏损的话,止损减少自己的损失;头寸出现盈利的话,就加仓扩大自己的盈利,这就是加仓的精髓.
- 海龟在突破时只建立一个单位的多头头寸，在建立头寸后以 1/2N的间隔增加头寸。这种 1/2N 的间隔以前面指令的实际成交价为基础。但对于单一市场最多可以购入4个单位。
- 对于股票市场，若股价在上一次买入（或加仓）的基础上上涨了0.5N，我们加仓一个Unit，但最多加仓到4Unit。

股票市场的加仓：

- 一般情况：若股价在上一次买入（或加仓）的基础上上涨了0.5N，我们加仓一个Unit，即假设N值仍为1，
  - ✓ 价格来到 10+ 1*0.5 =10.5时，加仓1个Unit，买入10000股，耗资10.5万元
  - ✓ 价格来到 10.5+ 1*0.5 = 11时，再加仓1个unit。买入10000股，耗资11万元
  - ✓ 价格来到 11+1*0.5=11.5时，再加仓1个unit，买入10000股，耗资11.5万元
- 购买上限：若股价上涨
  - ✓ 价格来到 11.5+1*0.5=12时，由于加仓已到4个unit，不再加仓。

止损：何时退出亏损的头寸

- 长期来看，不会止住亏损的交易员不会取得成功。关于止亏，最重要的是在你建立头寸之前预先设定退出的点位。
- 海龟以头寸风险为基础设置止损。任何一笔交易都不能出现2%以上的风险。因为价格波动 1N 表示 1%的帐户净值，容许风险为 2%的最大止损就是价格波动 2N。海龟的止损设置在多头头寸入市价格以下的 2N，空头头寸入市价格以 上的 2N。
- 我们当价格比最后一次买入价格下跌2N时，则卖出全部头寸止损

离市：何时退出赢利的头寸

- **何时退出赢利头寸的问题对于系统的收益性是至关重要的。任何不说明赢利头寸的离市的交易系统都不是一个完整的交易系统。**
- **海龟对于赢利头寸使用以突破为基础的离市策略，由于海龟系统有两个入市策略，因此也具备两种离市策略：**
  - 系统一离市对于多头头寸为 10 日最低价，对于空头头寸为 10 日最高价。如 果价格波动与头寸背离至 10 日突破，头寸中的所有单位都会退出。
  - 系统二离市对于多头头寸为 20 日最低价，对于空头头寸为 20 日最高价。如 果价格波动与头寸背离至 20 日突破，头寸中的所有单位都会退出。
- **我们采用系统一在股票市场入市，因此对于股票头寸为 突破10 日最低价，头寸中的所有单位都会退出。**

策略：如何买卖

- 信号一旦产生，关于执行策略考虑就变得重要起来。这对于规模较大的帐户尤其是个实际问题，因为其头寸的进退可能会导致显著的反向价格波动或市场影响
- 对于大多数的交易员，海龟系统离市或许是海龟系统法则中唯一最难的部分。等待 10 日或 20 日新低出现通常可能意味着眼睁睁地瞅着 20%、40%甚至 100%的可观利润化为泡影。
- 我们采用算法交易，一旦满足止损或离市的条件，则采用市价单在最短时间内离市。

海龟交易系统的代码及注释


```
#加载库函数
import numpy as np
import pandas as pd
import talib as ta                      # 加载技术分析库

#初始化回测环境
start = '20150201'                      # 回测起始时间 注：支持两种日期表述形式（ '2015-01-01'，'20150101'）
end = '20150601'                        # 回测结束时间
benchmark = 'SH50'                      # 策略参考标准为上证50指数
universe = set_universe('SH50')         # 选股范围为上证50成分股
freq = 'd'                              # 用日线回测的策略
refresh_rate = 1                        # 每1日调一次仓，即每个交易日都会运行第三部分的handle_data函数

#初始化投资者（账户）参数
#accounts为字典类型，代表投资者所有的账户，而字典中每一个键代表一个账户，而每一个键对应的值为该账户的初始情况，如本程序中的键为fantasy_account（股票账户），值为相应配置
accounts = {
    'fantasy_account': AccountConfig(account_type='security', capital_base=10000000)  #初始化投资者的股票账户： 投资品种为股票，初始投资金额为1千万
}

#初始化策略参数:
Max_Position_per = 0.1                  # 每只股票购入的最高比例为10%
max_history_window = 250                # 设置最长回测周期
Max_time_range  = 60                    # 设置数据回顾周期
limit_unit = 4                          # 限制最多买入的单元数
atrlength = 20                          # 计算真实波幅考虑的周期数
DC_range  = 20                          # 计算DC通道考虑的周期数
trade_percent = 0.01                    # 每次交易占总资产比例的基础值
record = pd.DataFrame({'symbol':[],'add_time':[],'last_buy_price':[]} )  #股票及对应的加仓次数和上一次买价

#初始化回测环境，指明创建账户时的工作，全局只运行一次
def initialize(context):
    pass

#handle_data函数是策略的核心函数，包含了所有策略算法的内容，包括数据获取，交易信号生成，订单委托等逻辑。
#handle_data函数无论是回测还是模拟交易场景，这个函数会根据回测频率 freq 的设置被调用。当freq='d'时，每天被调用一次，当freq='m'时，每分钟被调用一次。
def handle_data(context):
    timing_turtle(context)        # 基于海龟交易系统的择时策略
```

海龟交易系统的代码及注释


```
def timing_turtle(context):
    #全局变量声明
    global record                                                                    #声明record为局变量
    #数据获取（通用部分）：投资者账户，可供投资股票，价格数据，持仓数据，账户金额数据
    account = context.get_account('fantasy_account')                                #获取投资者的股票账户（fantasy_account）
    current_universe = context.get_universe(asset_type = 'stock',exclude_halt=True) #获取当前除停牌外的所有可供投资股票（universe）
    history = context.history(current_universe,['closePrice','lowPrice','highPrice'],Max_time_range, rtype='array')   #构建金肯特纳交易系统时考虑过去Max_time_range的历史数据
    security_position = account.get_positions()                                      #字典数据，上一K线结束后的有效证券头寸，即持仓数量大于0的证券及其头寸
    cash = account.cash                                                              #获取股票账户可用于投资的现金额度
    #按照海龟交易系统循环处理所有证券池中股票
    for sec in current_universe:                                                     #遍历所有可供投资股票；注：如该策略执行了选股策略，该部分应遍历被选中股票
    #数据获取（专有部分）：最低价，最高价，收盘价
        close = history[sec]['closePrice']                                             #获取股票sec过去Max_time_range天的收盘价
        low   = history[sec]['lowPrice']                                               #获取股票sec过去Max_time_range天的最低价
        high  = history[sec]['highPrice']                                              #获取股票sec过去Max_time_range天的最高价
        current_price = context.current_price(sec)                                     #获得当前时刻价格
    #数据处理（策略部分）：计算真实波幅，
        atr = ta.ATR(high, low, close, atrlength)[-1]                                  #计算真实波幅
    #策略部分(入场)：突破DC通道上轨，入场，买入1个单位的股票
        if  current_price> high[-DC_range:-1].max() and sec not in security_position:                  # 收盘价上穿DC上轨，且无持仓（突破通道上轨阻力位）
            unit = calcUnit(account.portfolio_value,atr)                                               # 计算建仓时应购入的股票数量
            order_to(sec,unit)                                                                         # 买入unit股的sec股票
            if len(record)!=0:
               record = record[record['symbol']!=sec]                                                  # 清空record中sec过期的记录
            record = record.append(pd.DataFrame({'symbol':[sec],'add_time':[1],'last_buy_price':[current_price]}))   # 记录股票，加仓次数及买入价格
            continue
    #策略部分(加仓)：若股价在上一次买入（或加仓）的基础上上涨了0.5N，则加仓1个单位的股票
        elif sec in security_position:                                                 # 判断是否已经持仓
            last_price =float(record[record['symbol'] == sec]['last_buy_price'])       # 上一次的买入价格
            add_price =last_price + 0.5 * atr                                          # 计算是否加仓的判断价格
            add_unit = float(record[record['symbol'] == sec]['add_time'])                  # 已加仓次数
            if current_price > add_price and add_unit < limit_unit:                        # 价格上涨超过0.5N并且加仓次数小于4次
               unit = calcUnit(account.portfolio_value,atr)                                # 计算加仓时应购入的股票数量
               order(sec,unit)                                                             # 买入1unit的股票
               record.loc[record['symbol']== sec,'add_time']=record[record['symbol']== sec]['add_time']+1   # 加仓次数+1
               record.loc[record['symbol']== sec,'last_buy_price']=current_price           # 加仓次数+1
    #策略部分(离场：止损或止盈)：当价格相对上个买入价下跌 2ATR时（止损）或当股价跌破10日唐奇安通道（止盈），清仓离场
            elif current_price< low[-int(DC_range/2):-1].min() or current_price < (last_price - 2*atr):
                order_to(sec,0)                                # 清仓离场
                record = record[record['symbol']!=sec]         # 将卖出股票的记录清空
```

海龟交易系统的回测结果

金程教育 GOLDEN FUTURE

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率	收益波动率	信息比率	最大回撤	换手率
1067.2%	157.8%	933.7%	0.84	19.64	54.2%	3.39	13.2%	0.31

回测详情　开始交易

专业来自101%的投入!

模块复用：投资历程的记录、查询与更新


```
#模块复用：投资历程的记录、查询与更新

#定义变量record，用以记录所有持仓股票的状态
record = pd.DataFrame({'symbol':[],'add_time':[],'last_buy_price':[]} )                    # 股票及对应的加仓次数和上一次买价

#在调用函数中，声明为全局变量
global record                                                                              # 声明record为局变量

#将新投资的股票信息记录到record中
 if len(record)!=0:
    record = record[record['symbol']!=sec]                                                 # 清空record中sec过期的记录
 record = record.append(pd.DataFrame({'symbol':[sec],'add_time':[1],'last_buy_price':[current_price]}))   # 记录股票，加仓次数及买入价格

#更新股票信息到record中
record.loc[record['symbol']== sec,'add_time']=record[record['symbol']== sec]['add_time']+1   # 加仓次数+1
record.loc[record['symbol']== sec,'last_buy_price']=current_price                          # 加仓次数+1

#卖出股票后，清空record中相关记录
record = record[record['symbol']!=sec]                                                     # 将卖出股票的记录清空
```

Thank you!