

# 多因子共振驱动的高频趋势突破策略

信弘量化交易大赛答辩 zkd047

奚项正 & 魏薇

少年班学院 & 数学科学学院，中国科学技术大学

2025 年 10 月 18 日

1. 队员介绍及职责分工
2. 设计方案介绍
3. 回测结果展示
4. 创新点及难点攻克
5. 未来展望

# 队员介绍及职责分工

## 队员 1：奚项正

独立负责策略的全部技术框架设计、核心算法实现与代码优化。

- 中国科学技术大学，少年班学院  
计算机科学与技术 & 金融学 (辅修)，本科，2023 年 9 月 - 至今

## 队员 2：魏薇

负责解读竞赛规则、管理项目时间线、确保所有交付物按时合规提交。从非技术视角，参与策略逻辑的讨论，提供反馈意见以完善策略。

- 中国科学技术大学，数学科学学院  
基础数学，硕士研究生，2024 年 9 月 - 至今
- 中国科学技术大学，数学科学学院  
数学与应用数学 (理学荣誉学士)，本科，2023 年 9 月 - 2024 年 6 月

## 1. 队员介绍及职责分工

## 2. 设计方案介绍

- 2.1 概述
- 2.2 指标计算
- 2.3 信号合成
- 2.4 仓位管理

## 3. 回测结果展示

## 4. 创新点及难点攻克

## 5. 未来展望

# 设计方案介绍 — 概述

交易策略文件 `strategy.py` 分为三个部分：

`generate_signals(...)` -> `pd.Series`:

核心策略逻辑函数，负责接收市场数据，并根据一系列复杂的规则生成具体的买卖仓位信号。— 指标计算 + 信号合成 + 仓位管理

`pred(date)`:

执行层函数，负责处理指定单个交易日的所有数据文件，调用核心策略函数进行计算，并保存结果。

`if __name__ == '__main__':`

程序主入口，负责获取所有待处理的交易日期，并利用多进程技术并行调用 `pred` 函数，以高效地完成所有数据的回测或预测任务。

# 设计方案介绍 — 概述

## 交易信号生成函数

```
def generate_signals(  
    df: pd.DataFrame,  
    # ... (策略参数)  
    bid_col: str = "BUYVOLUME01",  
    ask_col: str = "SELLVOLUME01",  
    high_col: str = "HIGHPRICE",  
    low_col: str = "LOWPRICE",  
    close_col: str = "LASTPRICE",  
    vol_col: str = "TRADEVOLUME",  
) -> pd.Series:
```

强化趋势确认  
参数微调 }  $\xrightarrow{\text{目标}}$  提高夏普比率

# 设计方案介绍 — 概述

- 核心策略参数:

window: `int` = 7200,: 滚动高点/低点和成交量均值的窗口大小  
ema\_period: `int` = 1200,: EMA (指数移动平均线) 的趋势过滤周期  
atr\_period: `int` = 240,: ATR (平均真实波幅) 的回溯周期, 用于动态风控

- 入场过滤参数:

vol\_multiplier: `float` = 1.8,: 突破时成交量倍数要求  
obi\_threshold: `float` = 0.15,: 盘口不均衡指标 (OBI) 阈值, 判断买卖盘压力  
use\_obi: `bool` = True,: 是否启用 OBI 作为额外过滤条件

- 风险管理参数:

stop\_loss\_mult: `float` = 1.2,: 动态止损, 关于 ATR 的倍数  
take\_profit\_mult: `float` = 3.5,: 动态止盈, 关于 ATR 的倍数  
time\_stop: `int` = 21600,: 最大持仓时间, 超时则强制平仓

# 设计方案介绍 — 指标计算

## 模块 0

```
data = df.copy()
required_cols = [high_col, low_col, close_col, vol_col]
if use_obi:
    required_cols.extend([bid_col, ask_col])
for col in required_cols:
    if col not in data.columns:
        print(f"Warning: Missing required column '{col}'. Filling with NaN.")
        data[col] = np.nan
```

确保关键列存在，如果不存在则填充 NaN 并发出警告。(鲁棒性!)



## 模块 1a: VWAP

```
cum_vol = data[vol_col].cumsum().replace(0, np.nan)
data["vwap"] = (data[close_col] * data[vol_col]).cumsum() / (cum_vol + 1e-12)
```

- 计算成交量的累计和
- 计算成交量加权平均价 (VWAP) — 被看作是市场参与者的平均持仓成本!

$$VWAP = \frac{(\text{价格} \times \text{成交量})\text{的累计和}}{\text{成交量的累计和}}$$

### 模块 1b: 滚动极值和成交量

```
data["rolling_high"] = data[high_col].shift(1).rolling(window).max()  
data["rolling_low"] = data[low_col].shift(1).rolling(window).min()  
data["vol_ma"] = data[vol_col].rolling(window).mean()
```

- 计算滚动窗口内的最高价，作为唐奇安通道的上轨
- 计算滚动窗口内的最低价，作为唐奇安通道的下轨
- 计算成交量的滚动平均值，作为判断成交量是否“激增”的基准

### 模块 1c: EMA 及其斜率

```
data["ema"] = data[close_col].ewm(span=ema_period, adjust=False).mean()  
data["ema_slope"] = data["ema"].diff().fillna(0)
```

- 计算收盘价的指数移动平均线 (EMA) — 越是久远的价格, 所占的权重就越小
- 计算 EMA 曲线的斜率 — 希望过滤掉趋势不明朗的震荡行情

$$\text{今日 EMA} = (\text{今日收盘价} \times \text{权重系数}) + (\text{昨日 EMA} \times (1 - \text{权重系数}))$$

### 模块 1d: ATR

```
high, low, close = data[high_col], data[low_col], data[close_col]
true_range = pd.concat([
    high - low, (high - close.shift(1)).abs(), (low - close.shift(1)).abs(),
], axis=1).max(axis=1)
data["atr"] = true_range.rolling(atr_period).mean().fillna(method='bfill')| .fillna(0)
```

- 计算真实波幅  $TrueRange = \max(H - L, |H - C_{prev}|, |L - C_{prev}|)$
- 计算平均真实波幅 (ATR) — 衡量市场的短期波动性, 影响后续动态止盈止损模块

### 模块 1e: OBI

```
if use_obi:
    data["obi"] = (data[bid_col] - data[ask_col]) / (data[bid_col] + data[ask_col] + 1e-12)
    data["obi"].fillna(0.0, inplace=True)
else:
    data["obi"] = 0.0
```

- 计算盘口不均衡指标 (OBI) — 基于市场微观结构的瞬时买卖压力

$$OBI = \frac{\text{买一量} - \text{卖一量}}{\text{买一量} + \text{卖一量}}$$

## 模块 2：最终开仓条件的信号合成

```
cond_long = (  
    (close > data["rolling_high"]) & (data[vol_col] > vol_multiplier * data["vol_ma"]) &  
    (close > data["ema"]) & (data["ema_slope"] > 0) & (close > data["vwap"]) &  
    (~use_obi | (data["obi"] > obi_threshold)))  
cond_short = (  
    (close < data["rolling_high"]) & (data[vol_col] > vol_multiplier * data["vol_ma"]) &  
    (close < data["ema"]) & (data["ema_slope"] < 0) & (close < data["vwap"]) &  
    (~use_obi | (data["obi"] < obi_threshold)))  
data["break_high"], data["break_low"] = cond_long, cond_short
```

将上述所有指标的判断逻辑予以整合，形成最终的开仓条件！

## 设计方案介绍 — 信号合成

- `(close > data["rolling_high"])`  
当前价格涨过了过去一段时间内的所有最高价格，可能预示新一轮上涨趋势
- `(data[vol_col] > vol_multiplier * data["vol_ma"])`  
市场上成交量远大于近期的均成交量，得到市场上大多数资金的认同和积极参与
- `(close > data["ema"])`  
当前价格处于其短期加权平均成本线 (EMA) 之上，顺应市场短期趋势
- `(data["ema_slope"] > 0)`  
要求 EMA 趋势线本身的方向也是向上倾斜的，在仍在发展的上升趋势中进行交易
- `(close > data["vwap"])`  
当前价格高于今天所有交易者的平均持仓成本 (VWAP)，市场情绪偏向乐观
- `(data["obi"] > obi_threshold)`  
买一价位上要显著多于卖一价位上的挂单数量，市场深度充足，买入意愿强烈

# 设计方案介绍 — 仓位管理

## 模块 3：带追踪止损的仓位管理

```
pos = np.zeros(len(data), dtype=np.int8)
current_pos, entry_price, entry_idx = 0, 0.0, -1
peak, trough = 0.0, 0.0
for i in range(len(data)):
    price = data[close_col].iloc[i]
    atr_val = data["atr"].iloc[i]
    # ... (仓位管理)
    pos[i] = current_pos
return pd.Series(pos, index=data.index, name="position").ffill().fillna(0)
```

策略执行的核心循环 — 逐根 K 线模拟交易过程，管理仓位的进入和退出。

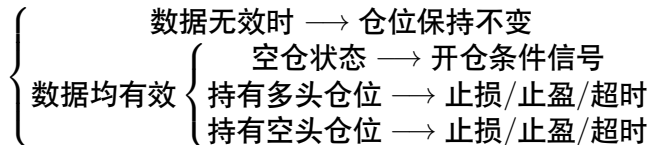


# 设计方案介绍 — 仓位管理

## 模块 3a

```
if not all(np.isfinite([price, atr_val, data["rolling_high"].iloc[i],  
    data["rolling_low"].iloc[i], data["vol_ma"].iloc[i], data["ema"].iloc[i],  
    data["ema_slope"].iloc[i], data["vwap"].iloc[i], data["obi"].iloc[i]])):  
    pos[i] = current_pos  
    continue
```

检查当前 K 线用于决策的所有指标值是否都是有效的有限数。(健壮性检查!)



### 模块 3b: 空仓状态 — 寻找开仓机会

```
if current_pos == 0:
    if data["break_high"].iat[i]:
        current_pos = 1
        entry_price, entry_idx, peak = price, i, price
    elif data["break_low"].iat[i]:
        current_pos = -1
        entry_price, entry_idx, trough = price, i, price
```

# 设计方案介绍 — 仓位管理

## 模块 3c: 持有多头仓位 — 管理止损止盈

```
elif current_pos == 1:
    peak = max(peak, price)
    sl = max(entry_price - stop_loss_mult * atr_val, peak - stop_loss_mult * atr_val)
    tp = entry_price + take_profit_mult * atr_val
    if price <= sl or price >= tp or (i - entry_idx >= time_stop):
        current_pos = 0
```

- 持有多仓期间的最高价 peak，持有空仓期间的最低价 trough — 实现追踪止损！
- 止损价 Stop Loss，控制单笔交易最大亏损，并一定程度上追踪保护已有浮动盈利
- 止盈价 Take Profit，设定合理盈利目标，系统化地将浮动盈利转化为实际盈利

价格触及止损  
价格触及止盈  
持仓时间超过上限

} 满足其一 → 平仓！

### 模块 3d: 持有空头仓位 — 管理止损止盈

```
elif current_pos == -1:
    trough = min(trough, price)
    sl = min(entry_price + stop_loss_mult * atr_val, trough + stop_loss_mult*atr_val)
    tp = entry_price - take_profit_mult * atr_val
    if price >= sl or price <= tp or (i - entry_idx >= time_stop):
        current_pos = 0
```

1. 队员介绍及职责分工
2. 设计方案介绍
3. 回测结果展示
4. 创新点及难点攻克
5. 未来展望

# 回测结果展示

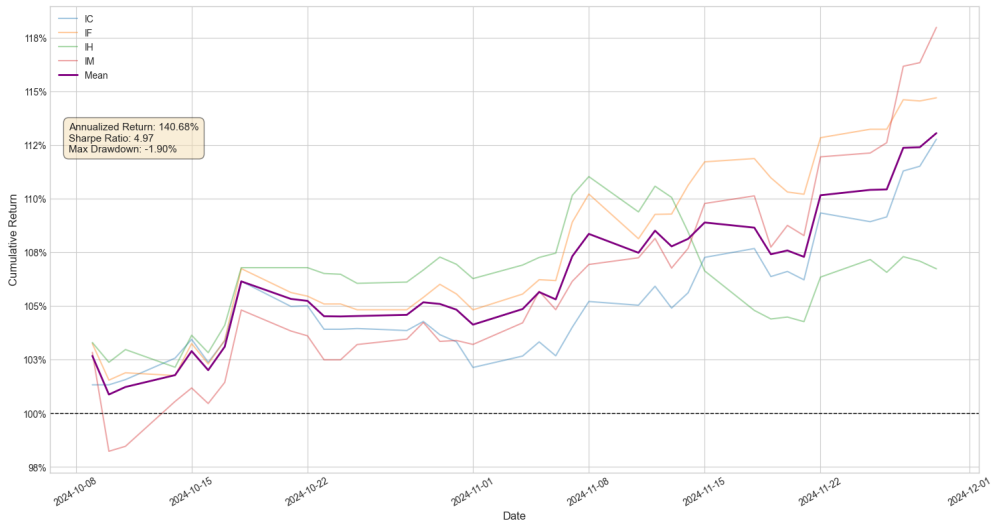


图: Cumulative Return Curve (Test Dataset)

# 回测结果展示

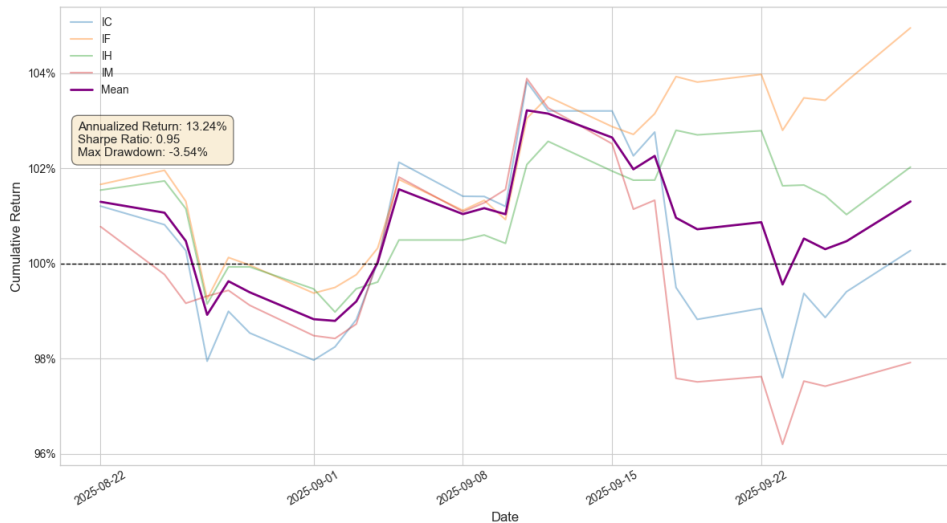


图: Cumulative Return Curve (Out-of-Sample Dataset)

# 回测结果展示



图: 30 Min K-line Chart with Equity and Positions (from 20241014 to 20241018)



1. 队员介绍及职责分工
2. 设计方案介绍
3. 回测结果展示
- 4. 创新点及难点攻克**
  - 4.1 策略目标解读
  - 4.2 我们面临的挑战
5. 未来展望

## 策略目标解读 (backtest.py)

```
annual_ret = all_rets_df['mean'].mean()* 252  
sharpe = all_rets_df["mean"].mean()/ all_rets_df['mean'].std()* np.sqrt(252)
```

$$Sharpe = \frac{\text{日组合收益率的均值}}{\text{日组合收益率的标准差}} \times \text{年化系数}$$

- 每个交易日内，每个可交易时间点上，对每个主力合约标的，可以在多或空方向上最多持仓 1 个单位。
- 当仓位变化即进行开仓/平仓操作时，会使用主力合约上下一个 Tick 的对手方一档价格作为成交价。
- 开仓/平仓操作的手续费皆按照 0.23‰ 计算；收盘时的非零仓位进行强制平仓。

# 创新点及难点攻克 — 策略目标解读

## 创新点一：多维确认的三重过滤入场机制

(我们不依赖单一信号，入场前必须通过三层连续检查。 — 日组合收益率的均值 ↗ )

- **检查价格行为**

确认价格突破的同时，成交量是否显著放大。

——> 确保信号的初始有效性，排除无量空涨。

- **检查趋势强度**

判断价格是否顺应 EMA 方向，并重点考察 EMA 斜率。

——> 只在趋势健康且在加速时入场，提高时效性。

- **检查日内情绪**

要求交易方向必须和 VWAP 所指示的主流资金方向一致。

——> 确保顺势而为，不和市场对着干。

## 创新点二：一个动态调整的风险管理系统

(我们的止损并非固定，会随着盈利增加而自动调整。 — 日组合收益率的标准差  $\searrow$  )

- 追踪盈利高点

系统会实时记录持仓期间的盈利最高点，并以此为基准来调整止损位。

- 检查趋势强度

当行情反转时，能保住大部分已赚到的利润，从机制上解决了盈利回吐的痛点。

- 优化盈亏比例

让盈利的单子尽可能多跑一会儿，是提升策略夏普比率的关键设计。

## 创新点三：逻辑驱动的严进宽出设计

- “严进”：严格的入场过滤

因为我们的三层确认标准非常严格，所以筛选出的交易机会本身就有很高的成功率。

- “宽出”：合理的容错空间

正因为入场质量高，我们可以设置一个相对更宽的初始止损。目的是给交易留出空间，来应对市场的正常波动，避免轻易被震出局。

- 系统更稳定！

这种设计降低了策略对某个特定参数的敏感度，让它在不同的市场环境下表现都更稳定。

# 创新点及难点攻克 — 我们面临的挑战

## 挑战一：信号与噪声的困境

这是策略研发中最核心的矛盾。我们发现：

- 如果过滤条件过于严苛，策略会变得非常迟钝，虽然能避开大部分噪声，但也因此错过了大量真实的盈利机会，导致长时间空仓，资金利用率极低。
- 如果过滤条件过于宽松，策略又会变得异常敏感，产生大量由市场随机波动引发的假信号，导致频繁交易。
- —→ 如何设计一套过滤体系，既能敏锐地捕捉到确定性的交易机会，又能强大到足以屏蔽市场的随机噪声，找到那个最佳的平衡点？

## 挑战二：交易成本的桎梏

在高频领域，成本是决定策略生死的隐形杀手。我们深刻体会到：

- 策略的单笔盈利预期往往非常微薄。即使在不考虑成本时是盈利的，高频次的交易也会让累积的手续费和滑点成本，最终吞噬掉所有的利润，甚至导致亏损。
- 这迫使我们不能满足于找到一个仅仅是方向正确的信号。我们必须找到那些预期盈利空间足够大、足以覆盖数个 Tick 成本的高质量信号。
- —→ 如何在保证交易频率的同时，确保每一笔交易的盈利预期都显著高于其执行成本？

## 挑战三：过拟合的陷阱

在拥有海量数据和众多可调参数的情况下，我们清楚地认识到过拟合的巨大风险。

- 我们很容易通过调整参数，在历史数据上雕刻出一条近乎完美的净值曲线。但我们知道，这往往是对历史噪声的过度学习。
- 一个策略真正的价值，不在于它过去表现多好，而在于它未来是否依然有效。
- —> 如何确保我们的策略逻辑是基于普适、可重复的市场行为规律，而非仅仅拟合了训练数据中的某些巧合，从而构建一个真正具备未来生命力的策略？



## 挑战四：跨合约信息的迷雾

我们曾尝试从更广阔的数据中寻找超额收益，但很快遇到了瓶颈。

- 理论上，非主力合约的资金流动、或者不同品种间的强弱分化，可能为主力合约的短期走势提供领先信息。
- 但在实践中，我们发现这些跨合约之间的关系往往非常不稳定且信噪比极低。今天看似有效的联系，明天可能就完全失效。
- —→ 如何从海量、嘈杂的多合约数据中，发掘出稳定、具备强因果性的领先信号，而非短暂或虚假的统计巧合？

1. 队员介绍及职责分工
2. 设计方案介绍
3. 回测结果展示
4. 创新点及难点攻克
5. 未来展望

	现状	未来展望
从规则到智能	依赖于固定的“与”逻辑和人工设定的阈值，决策相对刚性。	引入机器学习模型，对多维因子进行智能加权，实现从硬规则判断到柔性综合决策，以捕捉非线性规律。
从单点到组合	各合约独立生成信号，仅利用单一合约的时间序列信息，缺乏组合视角。	挖掘跨合约信息作为增强信号，并引入动态资金管理，构建分散化的策略矩阵，追求更平滑的组合收益。
从静态到动态	基于历史数据离线优化得到固定参数，对市场风格突变可能反应不够迅速。	探索参数自适应技术，让策略能根据近期市场状态自动微调其核心参数，提升策略鲁棒性。

# 谢谢大家！

## 参考资料



信弘量化交易大赛规则说明 2025.pptx



2025 比赛说明 - 飞书云文档

<https://nvz8qmeiid4.feishu.cn/docx/KzBUdG9OboqhsJx4d0RcRTA9ntf>