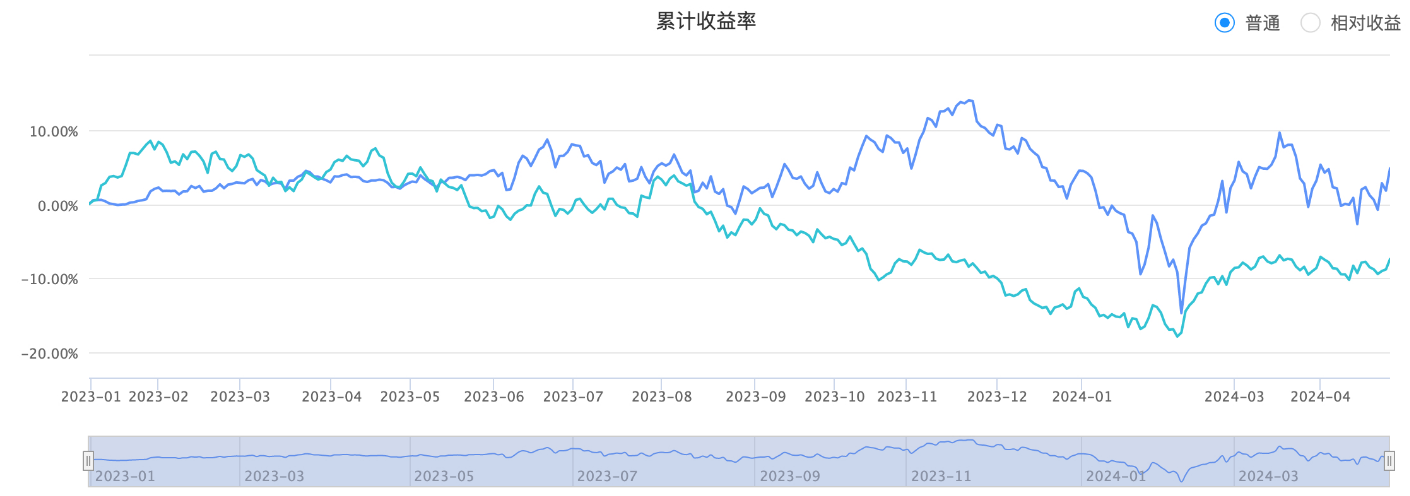
[2]:

```python
%%backtest
# from datetime import datetime
import talib

start = '2023-01-01'
end = '2024-04-28'
universe = StockUniverse('ZZ500')
benchmark = 'HS300'
freq = 'd'
refresh_rate = Monthly(1)
max_history_window = 60
fastperiod = 12  # 快线周期
slowperiod = 26  # 慢线周期
signalperiod = 9  # Signal平滑周期
# nstocks = 150

accounts = {
    'stock_account': AccountConfig(account_type='security', capital_base=1e5)
}


def initialize(context):
    pass


def handle_data(context):
    current_universe = context.get_universe(exclude_halt=True)
#     today = context.current_date
# #     print(today)
#     if today.strftime('%d') == '01':
#         prev_month_begin = today + pd.tseries.offsets.MonthBegin(-1)
#     else:
#         prev_month_begin = today + pd.tseries.offsets.MonthBegin(-2)
#     print('prev_month_begin:', prev_month_begin)
#     factor_exposure = get_data_cube(symbol=set_universe('A'),
#                                     field=['HSIGMA','VOL20'],
#                                     start=today, end=today,
#                                     style='ast').to_frame().reset_index()
    factor_exposure = context.history(symbol=current_universe, time_range=20, attribute=['HSIGMA','VOL20'], freq='d', style='tas',rtype='frame')
#     factor_exposure = pd.DataFrame(factor_exposure)
#     print(factor_exposure)
#     print(list(factor_exposure.keys()))
    factor_exposure = factor_exposure[list(factor_exposure.keys())[0]]
#     factor_exposure['HSIGMA'] = factor_exposure['HSIGMA'].to_numpy().reshape(-1,1)
#     factor_exposure['HSIGMA'] = pd.DataFrame(factor_exposure['HSIGMA'].tolist(),columns=['HSIGMA']).values
#     factor_exposure['VOL20'] = pd.DataFrame(factor_exposure['VOL20'].tolist(),columns=['VOL20']).values
#     display(factor_exposure)
#     #### Intersection #####
#     stocks_set = {}
#     stocks_set['rev'] = set(factor_exposure.sort_values('rev', ascending=True)['sec_id'].iloc[0:nstocks].tolist())
# #     stocks_set['mom'] = set(factor_exposure.sort_values('mom', ascending=False)['sec_id'].iloc[0:nstocks].tolist())
# #     stocks_set['illiq'] = set(factor_exposure.sort_values('illiq', ascending=False)['sec_id'].iloc[0:nstocks].tolist())
#     stocks_set['ivol'] = set(factor_exposure.sort_values('ivol', ascending=True)['sec_id'].iloc[0:nstocks].tolist())
#     current_universe = set.intersection(
#                             stocks_set['rev'],
# #                             stocks_set['mom'],
# #                             stocks_set['illiq'],
#                             stocks_set['ivol']
#     )
#     current_universe = list(current_universe)
    ##### Scoring #####
    cols = ['HSIGMA','VOL20']
#     for col in cols:
#         factor_exposure[f'{col}_rank'] = factor_exposure[col].rank()
    factor_exposure['HSIGMA_rank'] = factor_exposure['HSIGMA'].rank()
    factor_exposure['VOL20_rank'] = factor_exposure['VOL20'].rank(ascending=False)
    factor_exposure['rank_sum'] = factor_exposure['HSIGMA_rank'] + factor_exposure['VOL20_rank']
    current_universe = list(factor_exposure.sort_values('rank_sum').iloc[0:20].index)
#     print(current_universe)
#     factor_exposure['rev_rank'] = factor_exposure['rev'].rank()
#     factor_exposure['mom_rank'] = factor_exposure['mom'].rank(ascending=False)
#     factor_exposure['illiq_rank'] = factor_exposure['illiq'].rank(
#         ascending=False)
#     factor_exposure['ivol_rank'] = factor_exposure['ivol'].rank()
#     factor_exposure['rank_sum'] = factor_exposure['mom_rank']
# #                                     factor_exposure['rev_rank'] + factor_exposure['ivol_rank']
# #                                   factor_exposure['illiq_rank'] +
#     factor_exposure.sort_values('rank_sum', inplace=True)

#     current_universe = list(factor_exposure['sec_id'].iloc[20:40])
    history = context.history(current_universe, 'openPrice', 60, rtype='array')  # 拿过去60个交易日的开盘价来估算MACD

    account = context.get_account('stock_account')
    current_positions = account.get_positions()
    cash = account.cash
    buylist = []

    for stock in current_universe:
        close = history[stock]['openPrice']
        macd, signal, _ = talib.MACD(close, fastperiod=fastperiod, slowperiod=slowperiod, signalperiod=signalperiod)
        if macd[-2] < signal[-2] and macd[-1] > signal[-1] and stock not in current_positions:  # MACD上穿Signal, 且无持仓
            buylist.append(stock)
        elif macd[-2] > signal[-2] and macd[-1] < signal[-1] and stock in current_positions:  # MACD下穿Signal, 且有持仓
            account.order_to(stock, 0)  # 全部卖出
            cash += current_positions[stock].amount * context.current_price(stock)  # 估计买入金额

    d = min(len(buylist), int(cash) // 20000)  # 可以买入的股票数量, 如果资金不够, 只买入部分
    for stock in buylist[:d]:
        account.order(stock, 20000 / context.current_price(stock))
```

回测完成，耗时32.675秒

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率
3.8%	-5.9%	8.2%	0.84	0.04
收益波动率	信息比率	最大回撤	回撤恢复时间	年化换手率
20.8%	0.56	25.2%	--	57.89%



[3]:
```
# #查看调仓记录
show_order(start,end)
```

请重新回测后查看调仓记录

[4]:
```
# #查看持仓记录
show_position(start,end)
```

请重新回测后查看持仓记录

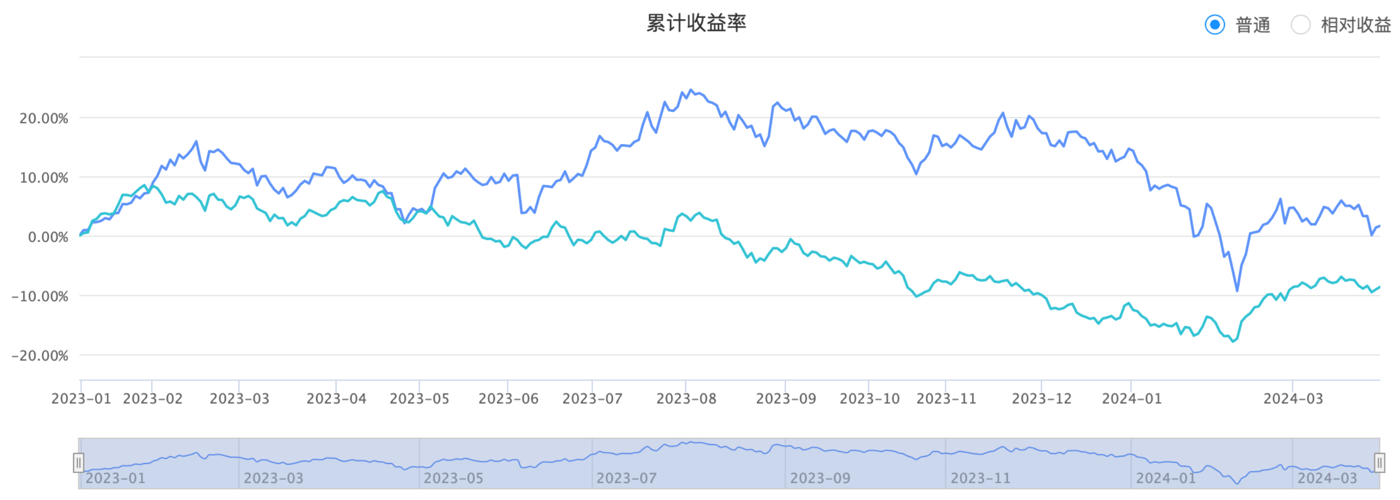
[8]:
```
%%backtest
# from datetime import datetime
import talib

start = '2023-01-01'
end = '2024-03-31'
universe = StockUniverse('ZZ1000')
benchmark = 'HS300'
freq = 'd'
refresh_rate = 1
max_history_window = 60
fastperiod = 12  # 快线周期
slowperiod = 26  # 慢线周期
signalperiod = 9  # Signal平滑周期
nstocks = 150

accounts = {
    'stock_account': AccountConfig(account_type='security', capital_base=1e5)
}


def initialize(context):
    pass


def handle_data(context):
    current_universe = context.get_universe(exclude_halt=True)
    today = context.current_date
#     print(today)
    if today.strftime('%d') == '01':
        prev_month_begin = today + pd.tseries.offsets.MonthBegin(-1)
    else:
        prev_month_begin = today + pd.tseries.offsets.MonthBegin(-2)
#     print('prev_month_begin:', prev_month_begin)
    factor_exposure = get_data_cube(symbol=current_universe,
                                field=['d6dk9218qq.size', 'd6dk9218qq.rev',
                                       'd6dk9218qq.illiq', 'd6dk9218qq.ivol'],
                                start=prev_month_begin, end=today,
                                style='ast').to_frame().reset_index()
#     print(factor_exposure)
    factor_exposure.columns = ['date', 'sec_id', 'size', 'rev', 'illiq', 'ivol']
#     assert factor_exposure['date'].nunique() == 1
    if factor_exposure.shape[0] == 0:
        pass
    else:
        factor_exposure = factor_exposure[factor_exposure['date'] == factor_exposure['date'].unique()[-1]]
        ##### Intersection #####
    #     stocks_set = {}
    #     stocks_set['rev'] = set(factor_exposure.sort_values('rev', ascending=True)['sec_id'].iloc[0:nstocks].tolist())
    # #     stocks_set['mom'] = set(factor_exposure.sort_values('mom', ascending=False)['sec_id'].iloc[0:nstocks].tolist())
    # #     stocks_set['illiq'] = set(factor_exposure.sort_values('illiq', ascending=False)['sec_id'].iloc[0:nstocks].tolist())
    #     stocks_set['ivol'] = set(factor_exposure.sort_values('ivol', ascending=True)['sec_id'].iloc[0:nstocks].tolist())
    #     current_universe = set.intersection(
    #                             stocks_set['rev'],
```

```python
    # #                               stocks_set['mom'],
    # #                               stocks_set['illiq'],
    #                               stocks_set['ivol']
    #     )
    #     current_universe = list(current_universe)
        ##### Scoring #####
        factor_exposure['rev_rank'] = factor_exposure['rev'].rank()
        factor_exposure['size_rank'] = factor_exposure['size'].rank(ascending=False)
        factor_exposure['illiq_rank'] = factor_exposure['illiq'].rank(
            ascending=False)
        factor_exposure['ivol_rank'] = factor_exposure['ivol'].rank()
        factor_exposure['rank_sum'] = factor_exposure['size_rank']
    #                                 factor_exposure['rev_rank'] + factor_exposure['ivol_rank']
    #                               factor_exposure['illiq_rank'] +
        factor_exposure.sort_values('rank_sum', inplace=True)

        current_universe = list(factor_exposure['sec_id'].iloc[20:40])
        history = context.history(current_universe, 'closePrice', 60, rtype='array')  # 拿过去60个交易日的收盘价来估算MACD

        account = context.get_account('stock_account')
        current_positions = account.get_positions()
        cash = account.cash
        buylist = []

        for stock in current_universe:
            close = history[stock]['closePrice']
            macd, signal, _ = talib.MACD(close, fastperiod=fastperiod, slowperiod=slowperiod, signalperiod=signalperiod)
            if macd[-2] < signal[-2] and macd[-1] > signal[-1] and stock not in current_positions:  # MACD上穿Signal，且无持仓
                buylist.append(stock)
            elif macd[-2] > signal[-2] and macd[-1] < signal[-1] and stock in current_positions:  # MACD下穿Signal，且有持仓
                account.order_to(stock, 0)  # 全部卖出
                cash += current_positions[stock].amount * context.current_price(stock)  # 估计买入金额

        d = min(len(buylist), int(cash) // 20000)  # 可以买入的股票数量，如果资金不够，只买入部分
        for stock in buylist[:d]:
            account.order(stock, 20000 / context.current_price(stock))

```

回测完成，耗时288.023秒

年化收益率	基准年化收益率	阿尔法	贝塔	夏普比率
1.4%	-7.2%	6.5%	0.79	-0.08
收益波动率	**信息比率**	**最大回撤**	**回撤恢复时间**	**年化换手率**
20.9%	0.48	27.2%	--	134.63%

累计收益率



[ ]: