# Winfor-Mkv

Mkv是为给定条件下，进行股票资产权重优化的计算而开发的python程序，理论基础是马尔科维茨的组合优化理论，适用于Windows 7及以上版本，Mac及Linus用户需要从源码进行重新打包，不能够直接运行 `Mkv_start.exe` 。

## 主要构成：

- **数据来源**：Wind提供的量化API，用户需要安装机构版Wind，并修复Python接口插件
- **参数设定**：用户可通过console窗口的提示依次输入所需要的参数，程序将输入的参数保存在 `configParam_mkv.conf` 文件。
- **优化器**：权重优化是一个quatratic optimization凸优化问题，使用的优化器是MOSEK，需要用户每年免费申请MOSEK license。许可证将发送到用户申请所提供的邮箱地址，下载 `mosek.lic` 文件到本地。
- **输入输出**：支持 `.txt .xls .xlsx` 格式输入需要回测的股票代码，输出结果采用 `.xlsx` 文件。

### 实现功能：

- [x] 指定在交易日交易时段内具体时刻$T$，提取股票或指数$S_i, i \in \{1,2,...,n\}$截止到定时$T$前$l$个交易日的日收益率$r_{it}, t \in \{T-l+1,...,T\}, i \in \{1,2,...,n\}$
  作为估计股票集合$S$日平均收益率向量$\boldsymbol{\mu}$和日收益率协方差矩阵$\boldsymbol{M}$的样本数据。
- [x] 支持部分约束：做空约束，组合年化波动率上限约束，单股最大绝对值权重约束。在约束条件下，最大化组合期望收益率。
- [x] *real-time* `mode=2` ：根据前$l$个交易日股票集合的收益率数据，优化结果$\boldsymbol{\omega}$作为下期持仓的建议。
- [x] *back-test* `mode=1` ：根据设定的开始时间 `start-time (yyyy-mm)` 和结束时间 `end_time (yyyy-mm)` 提取区间内的月末交易日日期$d_p, p \in \{1,2,...,P\}$，在每个月末时点上以前$l$个交易日的股票集合日收益率数据估计最优权重$\boldsymbol{\omega_p}$，并以此作为持仓权重持有资产至下月末$d_{p+1}$。在回测期内，将得到从第$2$月到第$P+1$月的模拟组合收益率。新版本支持不同频率（小时、日、周、月）四个频率上做的回测功能。
- [x] 实现条件选股功能，在全局环境 `global_spec` （依靠WindPy API支持，当前对A股市场支持最好。**TODO**：考虑其他交易所的数据可靠性和来源。）环境下，根据股票的基本面条件（依靠WindPy API支持：$P/E, ROE$等）初步筛选符合条件的股票，并且在GICS二级行业分类（ `ics` :*industry classification standard，可选变量，默认为GICS*）内按照初筛股票的市值（ `ics_fv` :*ics filter variable*，可选，目前推荐使用本文档列示的 `basic_indices` 可选内的变量，默认为市值；但也接受用户选择其他合法的wind变量代码。

### 模型说明：

首先说明变量与记号便于模型构建与理解。

- $r_{it}$：资产$i$在第$t$日的日收益率
- $c_t$：第$t$日的现金收益率，一般设定为常数$c$
- $\bar{x}_i$：资产$i$样本内日均收益率，$\bar{x}_i = \sum_{t=1}^{T} r_{it}$
- $\boldsymbol{M}$：资产集合$S$样本协方差
- $\boldsymbol{Q}$：$\boldsymbol{M}$的特征向量矩阵
- $\boldsymbol{\lambda}$：$\boldsymbol{M}$的特征值向量，所以有$\boldsymbol{M} = \boldsymbol{Q}\boldsymbol{\lambda}\boldsymbol{Q}^T$
- $\omega_i$：资产$i$在组合中的权重
- $u_i$：资产$i$的权重上限
- $\sigma$：组合年化标准差上限
- $N^{td}$：每年交易日天数
- $z$：目标函数，组合期望收益率

$$z = \sum_{i=1}^{n} \bar{x}_i \omega_i + (1 - \sum_{i=1}^{n} \omega_i)c = \sum_{i=1}^{n} (\bar{x}_i - c)\omega_i + c$$

这个二次规划问题的数学表达为：

$$max \sum_{i=1}^{n} (\bar{x}_i - c)\omega_i$$

$$s.t. \sum_{i=1}^{n} |\omega_i| \leq 1$$

$$\frac{1}{2}\boldsymbol{\omega}^T M \boldsymbol{\omega} \leq \frac{\sigma}{2\sqrt{N^{td}}}$$

$$|\omega_i| \leq u_i, for i \in \{1, 2, ..., n\}$$

需要提示的是，正则化条件要求特征向量$\boldsymbol{\lambda}$所有分量非负，否则 $\boldsymbol{M}$不满足半正定条件，此时采用 `nearPD.nearestPD()` 求出$\boldsymbol{M}$的“最近半正定矩阵”代替。

非半正定协方差矩阵在本例中不会产生于只能做多 `short=0` 的约束下，但如果允许做

空 `short=1` ，由于MOSEK的优化器表达不允许非线性的绝对值形式约束，因此需要将$\sum_{i=1}^{n}|\omega_i| \leq 1$进行先行改写。重定义资产$i$的权重：

- $\omega_i^+$：持有资产$i$的多头净权重
- $\omega_i^-$：持有资产$i$的空头净权重

因此有$\omega_i = \omega_i^+ - \omega_i^-$以及$|\omega_i| = \omega_i^+ + \omega_i^-$。将资产$i$的空头视作是另一个对偶资产$i'$，与资产$i$的收益率相关系数为-1，$r_{it} = -r_{i't}$。扩展的协方差矩阵$M' \in \mathbb{R}_{2n\times 2n}$不再是正定的，因此需要采用 `nearestPD()` 进行修正。

# 用户须知

## 1.文件及配置

用户使用本程序，可直接运行 `Mkv_main.exe` 可执行程序，不需要本地python环境，但是部分支持文件需要仔细配置。

a. 下载 `Mkv_main.exe` 到本地任意路径 ./directory/Mkv

b. 在同一文件夹下，新建 `WindPy.pth` 文件，并在文件中写入本机Wind安装地址，例如 `C:\Wind\Wind.NET.Client\WindNET\x64`

c. 将 `mosek.lic` 证书文件放置在此文件夹下

d. 进入Wind界面，在量化接口中修复python插件

e. 如果选择通过文件输入待回测或实盘组合优化的股票代码，请在 `.txt .xls .xlsx` 文件中，第一列写入需要进行回测的Wind股票代码

f. 下载 `configParam_mkv.conf` 到./directory/Mkv，直接在文件中修改对应的参数，注意注释中的解释和格式要求，并保存。

g. 双击 `Mkv_main.exe` 开始程序，程序将尝试从 `configParam_mkv.conf` 读入策略所需要的所有参数，输入参数不符合规范将导致程序报错。各个参数的释义及支持范围、填写格式请见“2.参数释义”部分。一个完整的 `configParam_mkv.conf` 文件填写实例请见"开发者说明-1.Mkv_main.py中的填写示例"

## 2.参数释义

**target_index**：字符串，目标追踪指数成分的指数wind代码（如恒生指数：HSI.HI）。该参数具有较高优先级，即如果同时输入该参数以及 `code_file` 参数，那么程序忽略 `code_file` 参数，而执行指数成分回测。

**code_file**:第e.步创建的包含目标股票资产的文件地址 eg. C:/Users/Dell/Mkv/codes.xls；如果以其他方式确定待回测股票，这一项不需填写。但是如果希望从文件中读入股票，那么应该保持 `target_index` 和 `global_spec` 两个参数为空。否则程序优先认为执行指数成分股回测或者基本面回测。

**work_file**:程序输出到的文件夹地址

**mode**:模式控制参数，如果*real-time* `mode=2` ；或者 *back-test* `mode=1`

**vol**：组合年化波动率上限，浮点数类型，eg 0.15，表示15%
**short**：表示做空约束， `short=1` 允许做空； `short=0` 仅能做多
**max_weight**：单股最大绝对值权重，浮点数类型
**cash_return**：年化现金收益率，浮点数类型
**frequency**：回测的频率，目前支持“H”，“D”，“W”，“M”（小时，日度、周度和月度）回测。日频及更低频回测，其回测基准K线都是日K线，区别是调仓频率不同。但小时频的基准K线是60min K线。Wind数据接口对于分钟级别数据使用上需要注意以下限制：

- wind的分钟级别数据仅包括大陆6大交易所，因此港股和美股无法采用分钟级别数据。
- 在分钟级别数据上应用低频的基本面数据意义不大，因此小时频回测暂支持 `code_file` 输入和 `target_index` 输入。
- 此外wind的分钟级别数据仅支持从当前时间开始向前3年内的数据，设置回测时间距离当前过远会导致错误。
- 由于当前WindPy API中w.wsi(提取分钟级别数据)版本调整导致数据读取不稳定，如果发现回测时出现大量Warning信息提示提取股票的数据失败（少量Warning信息提示是正常的，原因一般为该股票在该回测时间段内持续停牌，缺失该时间段全部数据），表示该数据接口连接不好。可能方法为关闭程序，重新启动尝试；或者更换时间尝试。

**rebalance_hour**：选择 `frequencey=H` 即小时级别K线为基本K线，该参数为整数，表示持仓周期的小时数。
**calc_time**：在 `mode=2` 下被调用，表示程序计算部分运行时间。期望是当日交易时间，如果早于当前，那么程序即刻运行，实际以当前时间为 `calc_time` ；如果晚于当前，那么等待至定时运行主程序。如果早于当日，那么认为是对自定义时点的单次回测；如果晚于当日日期，默认立即执行。输入请遵循格式yyyy-mm-dd HH:MM:SS, eg. 2018-12-10 10:20:00
**num_d**：用于回测的交易日长度
**start_time**：在 `mode=1` 下被调用，表示回测开始的年月。格式yyyy-mm，回测从该月末交易日收盘开始。
**end_time**：在 `mode=1` 下被调用，表示回测结束的年月。格式yyyy-mm，回测在该月末交易日收盘结束获得最后一次权重优化结果，并持有至后一个月结束。
**benchmark**：市场指数，作为组合表现基准，将被用于在输出结果时与组合期间收益率进行对比。
**global_spec**：采用基本面选股模式，字符串，表示全局股票范围，该参数优先级高于 `code_file` 但低于 `target_index` 。目前支持以下输入（大小写都可）；如果需要同时将多个股票空间合并进行筛选，可用;分隔（eg. HK;US_CSS表示全部港股和美股上市中概股）：

全部A股：A-SHARE
上证A股：SH
深圳A股：SZ
深圳主板A股：SZ_MAIN
深圳中小板：SZ_SME
深圳创业板：SZ_GE

香港全部股票：HK
香港恒生指数成分股：HK_HS
港股中资股(含H股、红筹股、中资民营股)：HK_CN
香港主板上市股票：HK_MAIN
美股全部股票：US
美股中概股：US_CCS

Tips:支持用户输入不在程序默认输入的wind板块代码。识别到用户输入以上列示 global_spec 以外参数，会输出Waring信息。用户查询需要板块的方法：

wind终端 -> 量化 -> API接口 -> 板块函数（WSEE） -> 板块查询 -> (查找所需板块“板块ID”字段) 。eg.进行以下搜索： 沪深股票 -> 香港股票 -> 港股市场类 -> H股 -> a002010500000000

**basic_indices**：采用基本面选股模式，字符串，表示若干个基本面变量条件，eg pe>=15 表示市盈率不低于15，不同条件之间用";"连接。目前支持的基本面变量如下（大小写皆可）。注意如果所有子行业都具体填写了相应筛选条件，那么本行可以缺省；此参数用于子行业筛选有缺省时的默认筛选条件：

市盈率TTM：pe
市值：ev
市净率：pb
市销率TTM：ps
市现率TTM：pcf
净资产收益率：roe
万德一致预期净利润同比：est_netprofit
分红收益率：dividendyield
净利润两年平均复合增长率：netprofit_cagr
资产负债率：debt_ratio
经营活动净现金流/经营利润TTM：ocf2op
30天平均收益率：amt30d

注意：1.所有类似市值的参数单位都是1（原始货币）；roe，dividendyield,est_netprofit,debt_ratio,netprofit_cagr,ocf2op表示为百分数（eg.如15%只需写15）。
2.est_netprofit,netprofit_cagr以及ocf2op等衍生指标在港股和美股可得数据中数据缺失值较多，可能难以用于筛选；在使用时应当注意。
3.接受用户输入不在程序默认常用指标，输入应当为合法wind指标。所有合法指标可通过以下方式进行查询：

wind金融终端 -> 量化 -> API接口 -> 代码生成器 -> 多维数据（WSS） -> 沪深股票 -> 市场类 -> 上证A股 -> (任选若干只股票) -> 下一步 -> (点开需要的指标，并将“字段”列示的wind指标名称作为参数传入) 。例如，点选 财务数据 -> 资本结构 -> 资产负债率 -> debttoassets 。

4.在选择指标时，可少量运行检查数据是否完整（选择好指标后， 下一步 -> 直接运行 ）。如果需要使用港股和美股数据，更应当检查数据是否有过多缺失值；在每次调整股票池时如果待选股票的指标存在缺失值，将被认为不满足条件。

**energy2**：表示GICS行业分类标准下“能源II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。

**materials2**：表示GICS行业分类标准下“材料II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**capital_goods**：表示GICS行业分类标准下“资本货物”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**commercial_prof_serv**：表示GICS行业分类标准下“商业和专业服务”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**transportation**：表示GICS行业分类标准下“运输”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**automobiles**：表示GICS行业分类标准下“汽车与汽车零部件”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**consumer_durables_apparel**：表示GICS行业分类标准下“耐用消费品与服装”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**consumer_serv2**：表示GICS行业分类标准下“消费者服务II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**media2**：表示GICS行业分类标准下“媒体II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**retailing2**：表示GICS行业分类标准下“零售业”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**food_retailing2**：表示GICS行业分类标准下“食品与主要用品零售II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**food_bev_tobacco**：表示GICS行业分类标准下“食品、饮料与烟草”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**household**：表示GICS行业分类标准下“家庭与个人用品”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**healthcare_equip_serv**：表示GICS行业分类标准下“医疗保健设备与服务”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**pharm_biotech**：表示GICS行业分类标准下“制药、生物科技与生命科学”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**banks**：表示GICS行业分类标准下“银行”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**diver_financials**：表示GICS行业分类标准下“多元金融”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**insurance2**：表示GICS行业分类标准下“保险II”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**software_serv**：表示GICS行业分类标准下“软件与服务”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。
**tech_hardware_equip**：表示GICS行业分类标准下“技术硬件与设备”行业的基本面筛选条件，语法格式同 basic_indices ，但是仅适用于该行业内的条件选股;缺省此行会采用 basic_indices 填补。

**semiconductors2**：表示GICS行业分类标准下“半导体与半导体生产设备”行业的基本面筛选条件，语法格式同 `basic_indices` ，但是仅适用于该行业内的条件选股;缺省此行会采用 `basic_indices` 填补。
**telecommun_serv**：表示GICS行业分类标准下“电信服务II”行业的基本面筛选条件，语法格式同 `basic_indices` ，但是仅适用于该行业内的条件选股;缺省此行会采用 `basic_indices` 填补。
**utilities2**：表示GICS行业分类标准下“公用事业II”行业的基本面筛选条件，语法格式同 `basic_indices` ，但是仅适用于该行业内的条件选股;缺省此行会采用 `basic_indices` 填补。
**real_estate2**：表示GICS行业分类标准下“房地产II”行业的基本面筛选条件，语法格式同 `basic_indices` ，但是仅适用于该行业内的条件选股;缺省此行会采用 `basic_indices` 填补。
**refresh_freq**：采用基本面选股模式，字符串，表示每隔多久刷新一次股票池。目前支持1M, 2M, 3M, 4M, 5M, 6M，分别表示每隔1个月到6个月。注意，该参数不受回测频率 `frequency` 影响，但是刷新次数与回测时间长度（ `start_time` ， `end_time` ）有关。
**ics**：采用基本面选股模式，字符串，二级行业分类标准。默认选用wind行业分类(industry_gics)。选择wind行业分类时，可适用子行业的不同筛选标准，当选择其他行业分类标准时，将为所有子行业适用统一筛选标准，参见 `basic_indices` 参数，用户输入的子行业筛选标准将被忽略。较常用的分类标准如下：

wind行业分类：industry_gics
申万行业分类：industry_sw
中信行业分类：industry_citic
国信行业分类：industry_gx
AMAC行业分类：indexname_AMAC

**ics_fv**：采用基本面选股模式，字符串，用于在子行业内决定排名先后的变量，默认为市值（ev）。其余可选排名变量请见**basic_indices**中所列示的变量。（**TODO**:解决硬编码基本面变量的问题）
**ics_rank**：采用基本面选股模式，表示选取子行业内按 `ics_fv` 排序后最终进入股票池的股票个数（不足该数的则全部入选）；支持3中输入方式；

- `n` -->输入单个整数n，则表示取排名前n的股票
- `n:m` -->输入索引切片形式，则表示取排名第n到第m的股票
- `[n,m,p,q]` -->输入列表形式，则表示取对应特定排名的股票

## 3.输出结果解释

结果将以 `.xlsx` 格式输出到设定的 `work_file` 文件夹下，命名包含相关参数设定。

### 3.1*back-test*模式

输出文件命名为*mkv_{code_file/target_index/global_spec}_{longOnly,short}_yyyymm-yyyymm_freq{M,W,D}.xlsx*，其中*code_file*为用户提供的输入文件名，*yyyymm*表示回测起始时间和结束时间。输出文件包含3+T张表：

- **sheet1:weights**：第一行code，表示股票代码；第二行name表示股票简称；之后各行表示对应频率末时点的优化权重结果。

- **sheet2:portfolio monthly return**：各频率末回测结果在下一个周期持仓组合的收益率表现。
- **sheet3:{num_d}-day mean returns**：第一行code，表示股票代码；第二行name表示股票简称；之后各行表示 在至每个月末时点过去 `num_d` 个交易日的股票的平均日收益率。用于检测优化结果表现。
- **sheet4:portfolio {daily, hourly} returns**：回测期间内，从第一次持仓直到回测结束的每个交易日滚动收益率。即表示从第$t-1$个交易日到第$t$个交易日的组合简单收益率。注意实际上，由于资产价格变化，在前一次调仓后到下一次调仓前，组合的权重都在不断变化。本表列示的收益率是考虑了权重变化后的收益率，即

$$p_t = \sum_{i=1}^{N} = w_{i,t} r_{i,t}$$

记每次调仓后的权重为$w_{i,0}$,在第t-1天收盘后，重新计算当天价格改变对于组合内各资产权重的更新：

$$w_t = \frac{w_{t-1} \odot (1 + r_{t-1})}{w_{t-1} \cdot (1 + r_{t-1})}$$

### 3.2*real-time*模式

输出文件命名为*mkv_{code_file}_{longOnly,short}_yymmdd.xlsx*，其中*code_file*为用户提供的输入文件名，*yymmdd*为 `calc_time` 的时间简写。输出文件包含3张表：

- **sheet1:weights**：第一列code，表示股票代码；第二列name表示股票简称；第三列weight,表示本次计算的优化权重结果。
- **sheet2:{num_d}-days mean returns**：第一列code，表示股票代码；第二列name表示股票简称；第三列mu表示本次计算中，各股票在 `calc_time` 时点前 `num_d` 个交易日期间的收益率均值。用于检测优化结果。

## 4.关于涉及日期的选用规范(NEW)

### 4.1*小时频回测*

小时频回测由于只涉及A股上市股票，日期上不会产生冲突，因此日期全部以深圳证券交易所的交易日历为标准。

### 4.2*日频回测*

日频级别较为复杂，由于日频级别回测的调仓周期为1d，考虑到如果选用工作日（正常周一至周五，不考虑节假日影响），将导致组合内资产在非交易日有调仓动作（实际上不能完成），因此仍采用**交易日**而非**工作日**日历。关于交易日历的锚定交易所，由以下标准决定：

- 默认交易所日历为：SZSE，即A股市场交易日历。

- 对于以下特殊情况，自动根据对应交易所日历进行操作：
  - 由文件输入股票的( `input_mode=3` )，如果组合内股票都在同一证券交易所上市，则自动采用该证券交易所日历锚定所有相关日期。
  - 盯住某一市场指数的( `input_mode=1` )，如果该市场指数成分股在同一交易所上市，则自动采用该指数对应交易所交易日历，如果该市场指数对应交易所不可得，则选用默认交易所日历。
  - 由global_spec( `input_mode=2` )确定股票范围的，根据wind对于板块编号的部分规律推定改板块对应交易所（之所以说“推定”，由于wind没有直接查询板块对应上市交易所的接口），板块wind代码满足一下条件的能够被正确对应到交易所，但多个spec组合若存在不同交易所则选择默认交易所日历：
    - 前6位为 `a00101` ，为A股上市，对应交易所日历SZSE
    - 前6位为 `a00201` ，为港股上市，对应交易所日历HKEX
    - 前6位为 `a00401` ，为台湾上市，对应交易所日历TWSE
    - 前6位为 `a00501` ，为美国上市，对应交易所日历NYSE
    - 大多数以 `10000` 开头的编号无特定规律，仅对于在 `SET_ID_DICT` (定义在 `Mkv_constant.py` 模块内)程序默认字典中出现的板块有正确编码。其余将使用默认交易所日历。

#### 4.3*周频、月频回测*

周频和月频回测涉及时间操作，大部分使用工作日（回测时间点，评估上一周期策略持仓收益时间点，组合和benchmark日频收益率序列）。但在每一个回测时间点上的前 `num_d` 个交易日收益率数据仍采用交易所日历。

# 开发者说明

当前程序运行需要python3.6及以上版本，其他python3版本需要修改字符串 `f'{var}'` 为 `{}.format{var}` 。Project包含5个.py文件，main函数入口在 `Mkv_start.py` 。Project运行需要载入以下packages: *configparser*, *os*, *openpyxl*, *xlrd*, *datetime*, *WindPy*, *numpy*, *re*, *time*, *dateutil*, *calendar*, *scipy*, *mosek*。

本项目代码提交至github该项目仓库 `master -> cods` 文件夹下；开发API文档请见 `master -> api/Mkv_API -> index.html` 。github本身不支持直接浏览html文件，请见 `api/Mkv_API` 文件夹clone到本地后再打开。鉴于模块细节可在API内查询，以下仅简单介绍本项目的基本组成。

## 1.Mkv_main.py

运行程序，创建 `Manage` 对象 `m=Manage()` ， `__init__()` 函数将执行参数默认赋值以及参数文件的 `configParam_mkv.conf` 的创建及读取工作，并提示用户输入参数并保存。

- `Manage.init_conf(self)` 在项目文件夹下尝试读取该配置文件，如果该文件不存在，将自动创建同名空白文件。因为参数涉及中文路径，因此需要采用gbk编码格式。

```
f = open(CONF_NAME, "a+")
f.close()
self.conf.read(CONF_NAME, encoding="gbk")
```

该函数还将检测配合文件的section和option是否完整并且补全。完整的参数文件应当包含至少：

```
[dir]
target_index =
code_file =
work_file = C:/Users/zhangchangsheng/Desktop/Mkv_output

[constraints]
mode = 1
vol = 0.15
short = 0
max_weight = 0.08
cash_return = 0.02

[calculation]
calc_time = 2018-12-05 09:33:00
num_d = 64
start_time = 2017-06-01
end_time = 2017-10-02
frequency = M

[performance]
benchmark = HSI.HI

[filter]
global_spec = HK_CN;a002010600000000
basic_indices = pe_ttm<=40;ev>=8000000000
refresh_freq = 1M
ics = industry_gics
ics_fv = ev
ics_rank = 2:8
energy2 =
materials2 =
capital_goods =
commercial_prof_serv =
transportation =
automobiles =
consumer_durables_apparel =
consumer_serv2 =
media2 =
retailing2 =
food_retailing2 =
food_bev_tobacco =
household =
healthcare_equip_serv =
pharm_biotech =
banks = pe>10;roe>10
diver_financials =
insurance2 =
software_serv =
tech_hardware_equip =
semiconductors2 =
telecommun_serv =
```

```
utilities2 =
real_estate2 =
```

这是配置文件初始化模板，用户可自行修改该文件或者通过运行程序输入参数修改和配置该文件。

- `Manage.set_code_file(self)` 用于从键盘读入文件地址，需要输入完整地址；如果无需更改之前的文件名，可以空白输入并Enter跳过。通过调用 `os.path.isfile` 检验是否存在该文件，如果文件有误则提示重新输入。
- `Manage.set_work_file(self)` 用于从键盘读入工作文件夹地址，该文件夹可以不存在， `os.path.makedirs` 将创建该文件夹并提示用户注意。最后一级文件夹命名 `/Mkv_output` 。如果无需更改之前的文件名，可以空白输入并Enter跳过。
- `Manage.set_mode` 设置运行模式， `mode=1` 表示*back-test*模式， `mode=2` 表示*real-time*模式。
- `Manage.set_time_interval` 设置在回测时间区间。在 `mode=1` 条件下被调用，输入格式为yyyy-mm。如果无需更改之前的设定，可以Enter跳过。
- `Manage.set_calc_time` 设置计算定时。在 `mode=2` 条件下被调用，输入格式为yyyy-mm-dd HH:MM:SS。根据用户输入，如果输入时间早于程序运行当天，以输入为节点进行历史回测；如果输入晚于程序运行当天，以当前收益率作为最后一个收益率观察值；如果输入时间是当日且晚于当前时刻，那么主程序计算等待至设定时间再运行；如果输入时间是当天但是早于当前时刻，那么以当前时刻为最后一个收益率观察时点立刻运行程序。如果无需更改之前的设定，可以Enter跳过。
- `Manage.set_cons_vol` 设置年化收益率约束。如果无需更改之前的设定，可以Enter跳过。
- `Manage.cons_up` 设置单只股票最大权重。如果无需更改之前的设定，可以Enter跳过。
- `Manage.set_cash_return` 设置现金的年化无风险收益率，默认是0.0。如果无需更改之前的设定，可以Enter跳过。
- `Manage.set_num_d` 设置用于估计股票两阶矩信息的样本长度，注意当前版本对未上市股票及停牌股票的收益率观察值都是0.0，后续开发需要注意辨别股票是否可进行交易的条件判定。。如果无需更改之前的设定，可以Enter跳过。
- `Manage.set_short` 设置做空约束， `short=1` 允许做空， `short=0` 不允许做空。
- `Manage.read_codes` 从文件中读取股票代码。调用 `Mkv_start.read_codes(f)` 读取文件，注意当前版本，仅支持.txt .xls .xlsx格式的输入文件，之后开发可以进行更广泛的扩展；并且当前版本中并没有添加对于输入合法性的检测，如果出现类似输入错误，eg. 60519.SH，正确应该为600519.SH；会在之后引发错误。
- `Manage.run_optimizer` 根据运行模式的设定，决定调用 `Manage.set_param{1,2}` 的时间。
- `Manage.set_params1` 在 `mode=1` 下运行，内部调用 `Mkv_data2.create_stocks, Mkv_data2.calc_params, Mkv_optimize2.optimizer` 进行创建股票对象、计算均值、协方差参数和进行优化的工作，并调用 `Manage.write_output1` 将返回的各资产包含现金的权重和相关参数写入输出文件中。
- `Manage.set_params2` 在 `mode=2` 下运行，内部调用 `Mkv_data2.create_stocks, Mkv_data2.calc_params, Mkv_optimize2.optimizer` 进行创建股票对象、计算均值、协方差参数和进行优化的工作，并调用 `Manage.write_output2` 将返回的各资产包含现金的权重和相关参数写入输出文件中。

- `Manage.get_backtest_t_seq` 根据设定的回测区间返回区间内各个月，调用 `WindPy.w.tdays` 获取月末日期

```
end_m = (datetime(*map(int, self.end_t.split("-")), 1) + relativedelta(months=1)).strftime("%Y-%
self.t_seq = w.tdays(self.start_t, end_m, "Period=M").Data[0]
self.t_seq = [t.strftime("%Y-%m-%d") for t in self.t_seq]
```

- `Manage.write_output1` 在 `mode=1` 下，将回测结果和回测产生的中间数据写入.xlsx文件。
- `Manage.write_output2` 在 `mode=2` 下，将回测结果和回测产生的中间数据写入.xlsx文件。

## 2.Mkv_data2.py

完成从数据库中提取数据并在进行优化前进行数据的预处理。文件定义了 `Stock` 类，该类对象包含5个fields： `code` 表示股票代码， `name` 表示股票简称， `mkt` 表示股票交易所， `r` 表示日收益率list， `t` 表示收益率序列的timestamp字符串list，格式为yyyy-mm-dd。该类对象作为储存股票raw data信息的数据对象。

- `Mkv_data2.ceate_stocks(codes, num_d, end=datetime.now().strftime("%Y-%m-%d"), q=True)` 其中参数 `codes` 表示股票代码字符串list， `num_d` 即为设定的样本数据时长， `end` 为本批样本数据结束时间， `q` 为是否是在盘中提取数据，如果是盘中提取数据，会调用 `WindPy.w.wsq` 提取当前收益率数据，否则调用 `WindPy.w.wsd` 提取历史日度收益率数据。返回值为以Stock对象为元素的list。
- `Mkv_data2.calc_params(stocks, short)` 参数 `stocks` 接受以Stock对象作为元素的list或其他可迭代对象， `short` 为布尔类型变量，指示做空约束条件。该函数用于计算股票的平均收益率，协方差矩阵，检验协方差矩阵的半正定性并进行必要的修正，将“修正后”的协方差矩阵转化为稀疏矩阵并按照非0元素的行列坐标储存其值。

```
mtx_r = [s.r for s in stocks]
mu = list(np.mean(mtx_r, axis=1).round(4))
cov = np.cov(mtx_r)
```

将list类型进行numpy.array类型的转换。如果是允许做空，还需要计算其扩展协方差矩阵并检验其正定性，检验方法是对其进行谱分解并判断是否有负特征值。

```
# 如果不通过正定性检验，需要进行remedy
cov1 = np.cov(np.concatenate((np.array(mtx_r),
-np.array(mtx_r))))
# 检验半正定性
eig1 = np.linalg.eigvals(cov1)
issd1 = bool(np.all(eig1 >= 0))
print("风险资产扩展矩阵是否半正定?" + str(issd1))
if not issd1:
    cov_pd = nearestPD(cov1)
```

将通过检验或修正后的协方差矩阵转化为稀疏矩阵并提取其下三角矩阵（因为对称性，可以只储存其一半信息）的行列坐标和值。

```
q = lil_matrix(cov_pd)
q_data = q.data
q_rows = q.rows
qsubi = []
qsubj = []
qval = []
for i in range(len(q_rows)):
    for j in range(len(q_rows[i])):
        if q_rows[i][j] <= i:
            qsubi.append(i)
            qsubj.append(q_rows[i][j])
            qval.append(q_data[i][j])
```

## 3.Mkv_optimize2.py

关于优化器具体使用，不在此赘述；当前版本设计优化的代码遵照MOSEK官方文档的一般格式，可以浏览该API文件：MOSEK optimiser api for Python。

## 4.Mkv_constant.py

定义了默认参数，将在用户输入部分失效的时候起作用。版本号，用于记录及后续更新。

## 5.建议

### 5.1 重新打包

完成新版本后的执行程序打包：cmd > `pip install pyinstaller`
进入项目所在文件夹> `cd directory/Mkv`
执行打包程序> `pyinstaller -F -c Mkv_start.py` 生成 `Mkv_start.exe` 文件在 `/directory/Mkv/dist/` 文件夹下。

### 5.2 README

不推荐在github上浏览README.md文件，鉴于其markdown浏览无法很好显示LaTex风格的数学公式。Atom对于Markdown的支持十分强大，通过 `apm install markdown-preview-enhanced` 插件下载，在编辑.md文件的时候可以实时浏览。在导出Markdown文件为其他格式时，可以通过 `apm install -g puppeteer` 插件将其另存为.pdf等格式。本文档推荐阅读pdf版本。

### 5.3 对API文档的持续维护更新

本项目采用 `sphinx` package生成html版本文档，提请注意在开发代码的同时尽量采用 `sphinx` 或其他API生成软件支持和建议的注释方式以减少后续调整工作。

- `sphinx` 官方文档请见：http://www.sphinx-doc.org/en/master/usage/restructuredtext/index.html
- `sphinx` 安装及配置可参考：https://www.jianshu.com/p/d4a1347f467b

### 5.4 关注github上issues讨论的历史

项目部分较复杂的改进细节会以issues的形式发布，在后续开发时请注意参考。