



quantstrat 實戰篇

林佳緯

程式交易實務 - 使用 R 語言 (三)



財團法人中華民國
證券暨期貨市場發展基金會
SECURITIES & FUTURES INSTITUTE

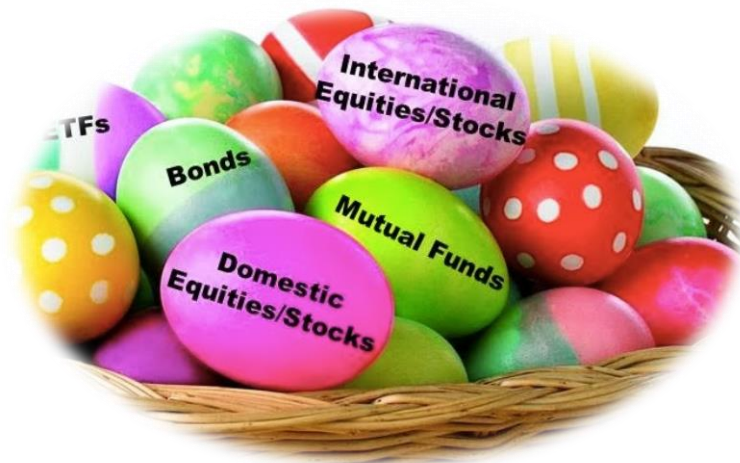
課程綱要

1. portfolio 多商品回測與rebalancing介紹
2. R 平行運算與效能調整技巧
3. 如何應用R程式交易套件於實際交易

投資組合理論基礎

投資組合理論

- 投資組合意義：
 - 指不同資產的組成，如果擁有資產為有價證券，即為證券投資組合。
 - 例如投資人以不同金額購買了股票、共同基金、債券、定期存單等資產。
- 現代證券投資組合理論
 - 馬可維茲博士提出
 - 定義統計上的標準差為風險測量值
 - 分散風險 別雞蛋放在一個籃子裡



現代投資組合理論

- 1952年諾貝爾經濟獎得主：亨利·馬可維茲(Harry Markowitz)，在其投資組合的選擇書中，首先提及風險應該以整個投資組合的風險來看，強調要追求較高的報酬率，就無法避免承擔風險
- 並說明如何建稱最佳投資組合，即在特定風險水準上，預期報酬率最高的投資組合
- 引導出投資商品除了**風險與報酬**之外的第三個變數—**相關係數**。
且 $-1 \leq \text{相關係數} \leq 1$ 。

相關的名詞定義

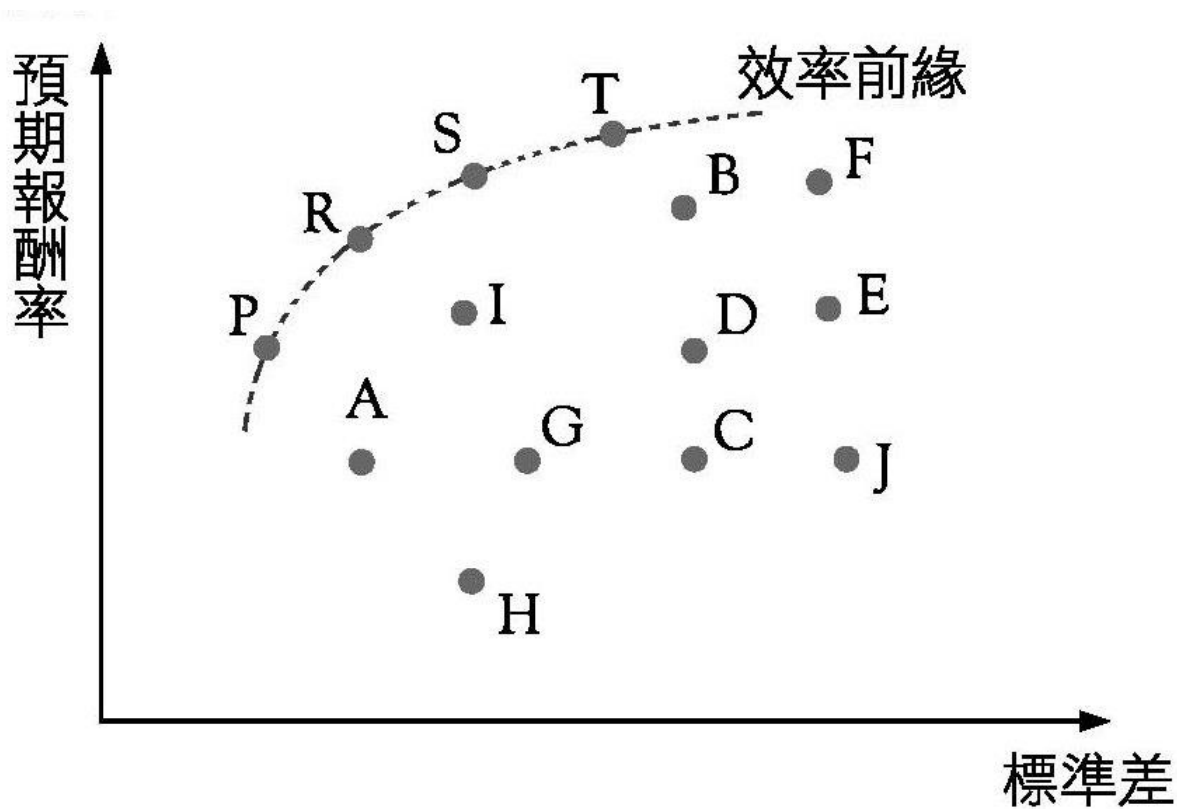
- 偏好(preference)
 - › 投資人對於風險及報酬率的態度，學術界稱之為「偏好」。
- 效用(utility)
 - › 投資人會選擇可以讓自己得到最大滿足感的投資組合，滿足感的學術專有名詞為「效用」。
 - › 基本假設是報酬率愈高，風險愈小，投資人的效用愈大。效用是報酬率與風險所決定的，報酬率對效用有正面影響，風險對效用有負面影響。
- 無差異曲線(Indifference curve)
 - 一種用來表明消費者對於兩種財貨，在各種不同數量的組合下，都有相同效用水準的軌跡。其特性有：負斜率、凸向原點、任兩條無差異曲線不相交，以及離原點越遠，效用越大。

相關的名詞定義

- 標準差(Standard Deviation)
 - › 表示分散程度的統計觀念，為各變量離均差平方的算數平均數的方根，標準差其實是一種平均距離的觀念，如果未來可能發生的報酬率會離平均報酬率很遠，就表示風險很大。目前被廣泛的運用在投資風險的衡量上，標準差是風險的量化指標。
- 投資決策法則
 - › 當風險相同時，選擇期望報酬較高者，而當期望報酬率一樣時，選擇風險較低者，這個法則稱為期望報酬-變異數法則。
- 效率前緣(Efficient Frontier)
 - › Markowitz 以期望報酬-變異數分析法，在符合投資決策法則情況下，運用統計分析與線性規劃技巧，告訴我們如何由一組資產之中選取最適投資組合。根據這樣理念建立所謂效率前緣(efficiency frontier)，作為報酬率與承受風險的最高邊界。

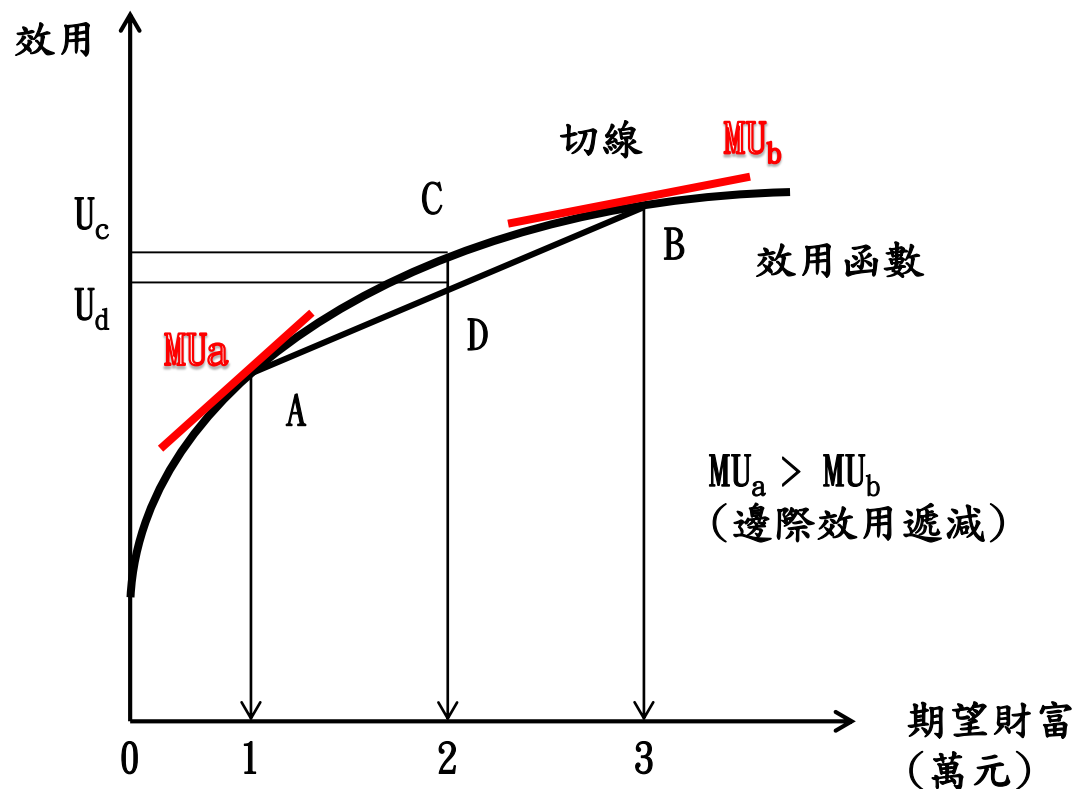
效率前緣

- 預期報酬與變異數間最適的投資組合前緣線



風險趨避特性

- 標準差(Standard Deviation)



A點到B點斜率斜率

$\frac{dU}{dE(W)}$ 遞減(MU_a 至 MU_b 逐漸變小)

代表「邊際效用(Marginal Utility)」隨期望財富遞減

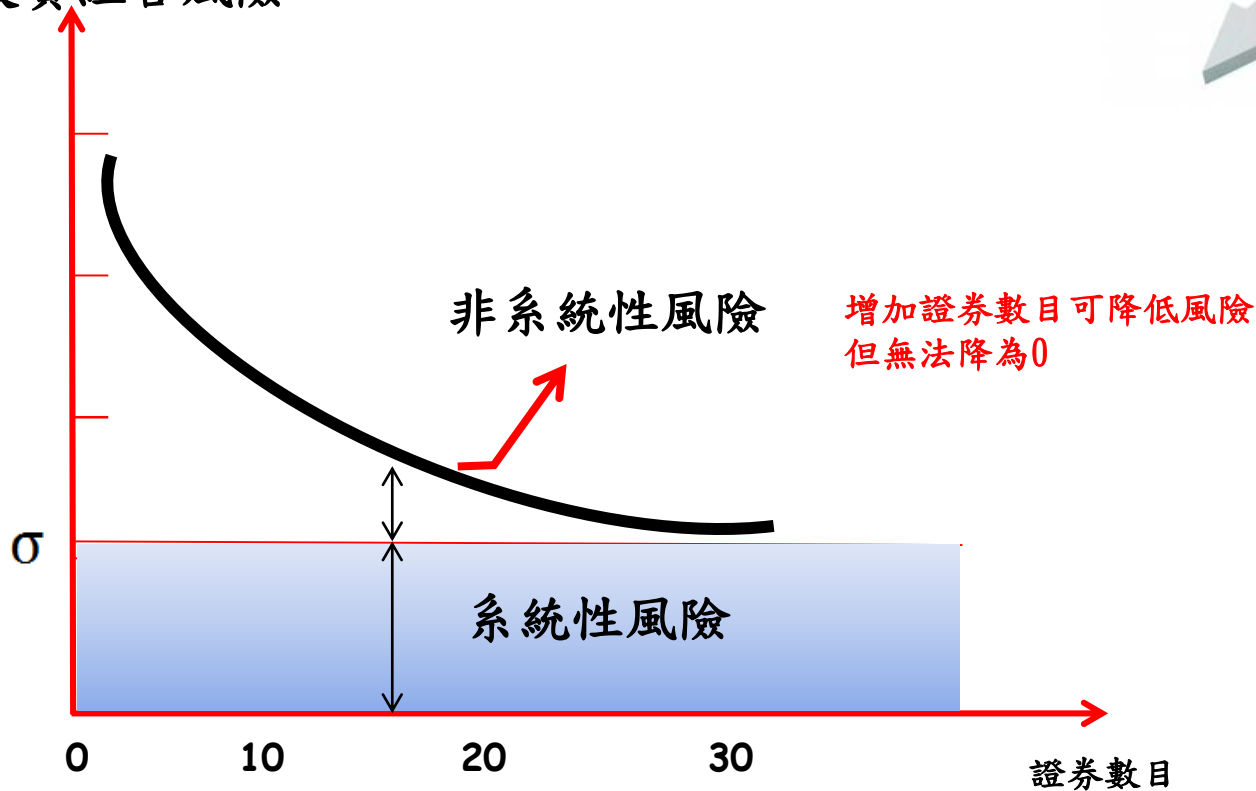
效用函數二次微分 $\frac{d^2U}{d[E(W)]^2} < 0$

意味這種投資人為厭惡風險者

投資組合與系統風險

- 證券投資組合因證券數目增加，會使投資風險降低
券才能達到降低風險的目的呢？

投資組合風險



R 投資組合最佳化套件

實作程式碼來源： Ross Bennett - R/Finance 2014
Complex Portfolio Optimization with PortfolioAnalytics

最佳化相關處理套件

- 均值-變異數投資組合最佳化
 - 二項式規劃
 - `tseries`, `quadprog`, `portfolioAnalytics` 等套件
- 條件性風險值(CVaR)最佳化
 - 線性規劃
 - `Rglpk`套件
- 一般性非線性最佳化
 - 差分演化演算法
 - `DEoptim`套件

R PortfolioAnalytics 套件

- 設計用來提供投資組合最佳化所需數值分析與視覺化圖表，能處理許多複雜之限制項與目標函數。
 - 提供多限制項與目標型態
 - 目標函數可以是任何R的函數
 - 模組化限制項與目標
 - 支援使用者自訂的動差函數
 - 視覺化圖表
 - 支援平行處理

PortfolioAnalytics 支援求解

- 線性與二項式規劃求解法
 - R Optimization · Infrastructure (ROI)
 - GLPK (Rglpk)
 - Symphony (Rsymphony)
 - Quadprog (quadprog)
- 全域(隨機或連續求解法)
 - Random Portfolios
 - Differential Evolution (DEoptim)
 - Particle Swarm Optimization (pso)
 - Generalized Simulated Annealing (GenSA)

投資組合設定參數

- portfolio.spec

```
## function (assets = NULL, category_labels = NULL, weight_seq = NULL,  
## message = FALSE)  
## NULL
```

- 起始化投資組合資料物件

- 存入商品歷史資料
- 存入限制式
- 存入條件式

投資組合設定參數

- add.constraint

```
## function (portfolio, type, enabled = TRUE, message = FALSE, ...,  
## indexnum = NULL)  
## NULL
```

- 限制條件種類：

- Sum of Weights
- Box
- Group
- Factor Exposure
- Position Limit
- 其它

投資組合設定參數

- add.objective

```
## function (portfolio, constraints = NULL, type, name, arguments = NULL,  
## enabled = TRUE, ..., indexnum = NULL)  
## NULL
```

- 目標函式種類：
 - Return
 - Risk
 - Risk Budget
 - Weight Concentration

執行投資組合優化

- optimize.portfolio

```
## function (R, portfolio = NULL, constraints = NULL, objectives = NULL,  
## optimize_method = c("DEoptim", "random", "ROI", "pso", "GenSA"),  
## search_size = 20000, trace = FALSE, ..., rp = NULL, momentFUN =  
"set.portfolio.moments",  
## message = FALSE)  
## NULL
```

- optimize.portfolio.rebalancing

```
## function (R, portfolio = NULL, constraints = NULL, objectives = NULL,  
## optimize_method = c("DEoptim", "random", "ROI"), search_size = 20000,  
## trace = FALSE, ..., rp = NULL, rebalance_on = NULL, training_period = NULL,  
## trailing_periods = NULL)  
## NULL
```

投資組合分析結果

VISUALIZATION

plot

chart.Concentration

chart.EfficientFrontier

chart.RiskReward

chart.RiskBudget

chart.Weights

DATA EXTRACTION

extractObjectiveMeasures

extractStats

extractWeights

print

summary

投資組合隨機樣本產生

- 三種不同產生方法：
 - 以 Pat Burns的取樣法 為基礎
 - 以 W. T. Shaw的單純型法(Simplex Method) 為基礎
 - 以 NMOF 套件裡的 gridSearch 函數 為基礎










資料準備

```
setwd("C:/證基會課程/Day 3/practices")
```

```
source("data_prep.R")
```

```
library(PortfolioAnalytics)  
require(doParallel)
```

```
cl <- makeCluster(4)  
registerDoParallel(cl)
```

Data		
edhec	An 'xts' object on 1997-01-31/2014-01-31 containing:	
equity.data	An 'xts' object on 1997-01-07/2010-12-28 containing:	
largecap_weekly	An 'xts' object on 1997-01-07/2010-12-28 containing:	
market	An 'xts' object on 1997-01-07/2010-12-28 containing:	
microcap_weekly	An 'xts' object on 1997-01-07/2010-12-28 containing:	
midcap_weekly	An 'xts' object on 1997-01-07/2010-12-28 containing:	
R	An 'xts' object on 1997-01-31/2014-01-31 containing:	
Rf	An 'xts' object on 1997-01-07/2010-12-28 containing:	
smallcap_weekly	An 'xts' object on 1997-01-07/2010-12-28 containing:	

Portfolio 隨機樣本產生

```
##### 投資組合條件設定 #####
stocks <- colnames(equity.data)

# 起始化條件
portf.init <- portfolio.spec(stocks)
# Add constraints
# 加權總數為 1
portf.dn <- add.constraint(portf.init, type="weight_sum",
                           min_sum=-0.01, max_sum=0.01)

# 箱型 限制式
portf.dn <- add.constraint(portf.dn, type="box",
                           min=-0.2, max=0.2)
portf.dn <- add.constraint(portf.dn, type="position_limit", max_pos=20)

betas <- t(CAPM.beta(equity.data, market, Rf))

portf.dn <- add.constraint(portf.dn, type="factor_exposure", B=betas,
                           lower=-0.25, upper=0.25)

rp <- random_portfolios(portf.dn, 10000, eliminate=TRUE)

view(rp)
```

Portfolio 隨機樣本產生

按照情境與限制條件，產生的樣本資料

	ORCL	MSFT	HON	EMC	DELL	KO	DD	XOM	GE	IBM	PEP	MO	COP	AMGN	CVX	TROW	TECD	FO
1	0.000	0.000	-0.050	0.000	0.000	0.000	0.000	0.000	0.198	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.044	0.000	-0.174	0.000	0.000	0.000	0.138	0.008	0.000	0.000	0.000
3	-0.032	0.000	0.000	0.000	0.000	0.000	0.000	-0.078	0.000	0.000	0.170	0.134	0.000	-0.098	0.000	0.000	0.000	0.000
4	-0.020	0.000	0.000	-0.052	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	0.158	0.000	-0.170	0.000	0.000	0.000	0.000	0.000	0.158	0.098	-0.040	0.072	-0.106	0.000	0.000	0.000	0.184	0.000
6	-0.024	0.098	-0.026	-0.064	-0.162	0.070	0.000	0.000	0.000	0.092	0.000	0.000	-0.174	-0.064	-0.032	0.000	0.122	-0.020
7	0.000	0.000	0.000	-0.122	0.188	0.058	0.000	0.000	0.000	-0.034	-0.152	0.000	-0.102	0.158	0.020	-0.160	0.000	0.000
8	0.000	0.150	0.114	-0.178	-0.094	0.000	-0.036	0.000	-0.098	0.000	-0.200	0.076	-0.002	-0.066	-0.092	-0.026	0.000	-0.100
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.000	0.000	0.000	-0.112	-0.188	0.186	0.000	0.000	0.000	0.000	0.030	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.000	-0.098	0.192	0.000	-0.012	0.000	0.000	0.000	-0.192	0.000	0.000	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	-0.058	-0.086	0.000	0.000	0.000	0.060	0.074	0.000	0.000	0.000	-0.096	-0.196	0.000	0.000	0.000	0.000
13	0.000	0.000	-0.042	-0.196	0.000	0.000	0.092	0.000	0.000	-0.008	0.000	0.000	0.000	0.084	0.000	0.000	0.040	0.000

Showing 1 to 14 of 9,868 entries

市場中立策略範例

- 我們假設設計一個股票投資組合部位：
 - 目標：
 - 極大化投報率：0.0015
 - 極小化風險(StdDev)：0.02
 - 限制：
 - 保持貨幣中立(Dollar Neutral)
 - 權重上下限區間(Box)：-0.2 ~ +0.2
 - 股票市場指數相關性(Beta)： -0.25 ~ +0.25
 - 有部位限制的風險管理： 最多20單位

市場中立策略範例

- 條件起始化設定

```
library(PortfolioAnalytics)  
require(doParallel)
```

```
cl <- makeCluster(4)  
registerDoParallel(cl)
```

```
# 商品契約的歷史資料準備  
source("data_prep.R")
```

```
portf.dn <- portfolio.spec(stocks)
```

市場中立策略範例

- 設定限制式

```
##### 設定限制式 #####
```

```
# Add constraint such that the portfolio weights sum to 0*  
portf.dn <- add.constraint(portf.dn, type="weight_sum",  
                           min_sum=-0.01, max_sum=0.01)
```

```
# Add box constraint such that no asset can have a weight of  
greater than  
# 20% or less than -20%  
portf.dn <- add.constraint(portf.dn, type="box", min=-0.2,  
                           max=0.2)
```

```
# Add constraint such that we have at most 20 positions  
portf.dn <- add.constraint(portf.dn, type="position_limit",  
                           max_pos=20)
```

```
# Add constraint such that the portfolio beta is between -0.25  
and 0.25  
betas <- t(CAPM.beta(equity.data, market, Rf))  
portf.dn <- add.constraint(portf.dn, type="factor_exposure",  
                           B=betas, lower=-0.25, upper=0.25)
```

市場中立策略範例

- 設定目標函式

```
##### 設定目標函式 #####
```

```
# Add objective to maximize portfolio return with a target of  
0.0015
```

```
portf.dn.StdDev <- add.objective(portf.dn, type="return",  
name="mean", target=0.0015)
```

```
# Add objective to minimize portfolio StdDev with a target of  
0.02
```

```
portf.dn.StdDev <- add.objective(portf.dn.StdDev, type="risk",  
name="StdDev", target=0.02)
```

市場中立策略範例

- 執行最佳化

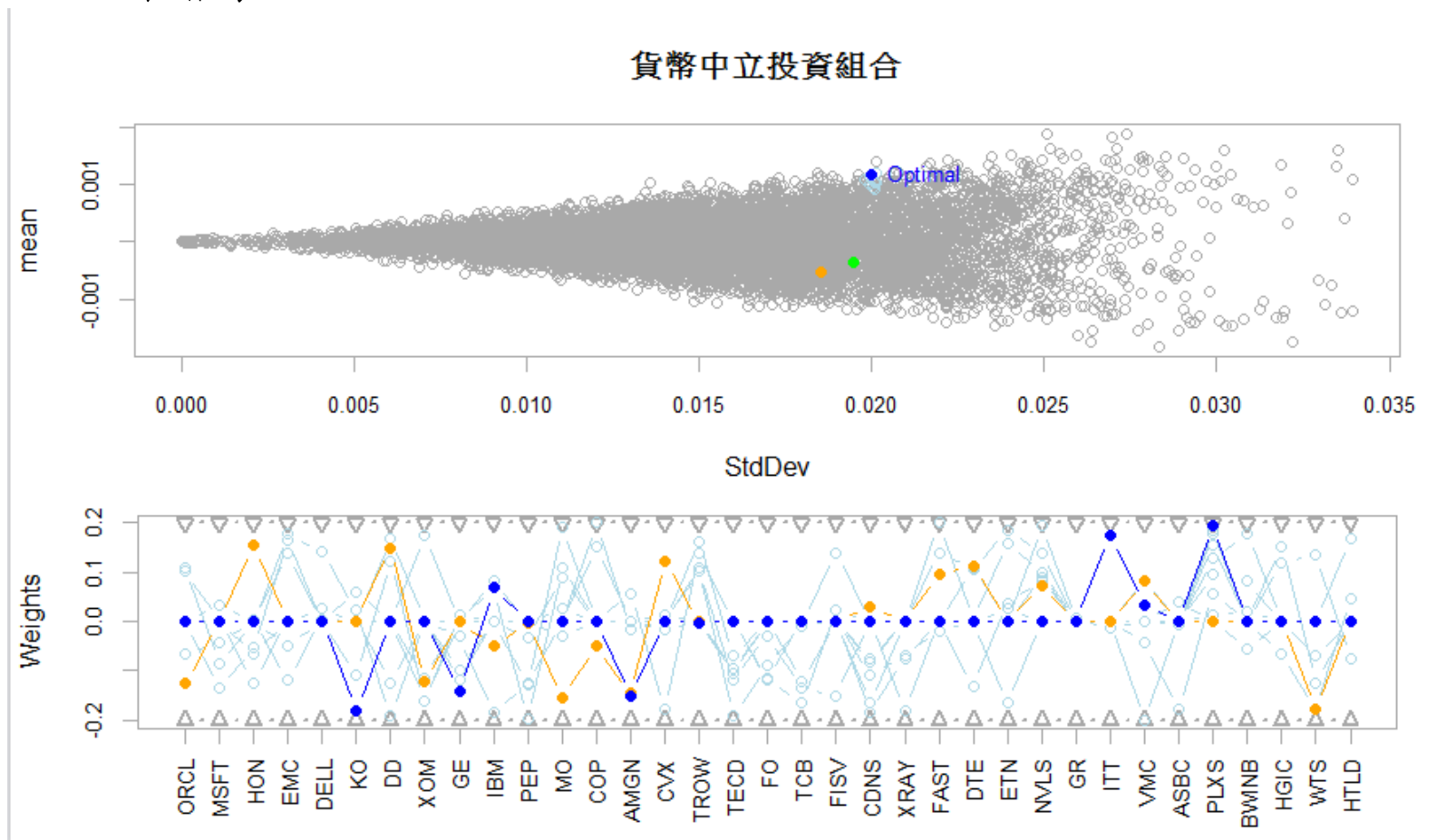
```
##### 執行最佳化 #####  
  
# Generate random portfolios  
rp <- random_portfolios(portf.dn, 10000, "sample")  
  
# Run the optimization  
opt.dn <- optimize.portfolio(equity.data, portf.dn.StdDev,  
                             optimize_method="random", rp=rp,  
                             trace=TRUE)
```

- 結果報表

```
##### 結果報表 #####  
  
plot(opt.dn, main="貨幣中立投資組合", risk.col="StdDev", neighbors=10)
```

市場中立策略範例

- 結果報表



R 平行運算與效能調整技巧

平行運算套件差異

- doMC (架構在multicore套件上, 適用於unix-相容作業系統)
 - doSNOW (架構在snow套件上, 只適用於Windows)
 - doParallel (架構在parallel上, 適用於上述兩種)
-
- parallel 套件融合了multicore 與 snow 基本重要的功能, 而且自動配置適合作業系統, 但Windows上只支援單一核心。

簡單暖身實例

```
library(doParallel)

detectCores()
## [1] 4
# Create cluster with desired number of cores
cl <- makeCluster(3)

# Register cluster
registerDoParallel(cl)

# Find out how many cores are being used
getDoParWorkers()
```


Foreach 單核心比較

- 單顆核心計算

```
require(foreach)
require(doParallel)
```

```
cl <- makeCluster(4)
registerDoParallel(cl)
```

```
# Mac 用下面套件
# require(doMC)
# registerDoMC(cores=1)
```

```
system.time(m <- foreach(i=1:100) %do%
{ matrix(rnorm(1000*1000), ncol=5000); NULL } )
```

```
system.time(m <- foreach(i=1:100) %dopar%
{ matrix(rnorm(1000*1000), ncol=5000); NULL } )
```

Foreach 多核心比較

- 使用四個核心平行計算

```
require(foreach)
require(doParallel)
```

```
cl <- makeCluster(4)
registerDoParallel(cl)
```

```
# Mac 用下面套件
# require(doMC)
# registerDoMC(cores=1)
```

```
system.time(m <- foreach(i=1:100) %do%
{ matrix(rnorm(1000*1000), ncol=5000); NULL } )
```

```
system.time(m <- foreach(i=1:100) %dopar%
{ matrix(rnorm(1000*1000), ncol=5000); NULL } )
```

Foreach 結果資料處理

- 資料回傳為向量

```
foreach(i = 1:5, .combine='c') %do% sqrt(i)
```

```
[1] 1.000000 1.414214 1.732051 2.000000 2.236068
```

- 資料回傳為table欄位

```
foreach(i = 1:5, .combine='cbind') %do% sqrt(i)
```

```
      result.1 result.2 result.3 result.4 result.5  
[1,] 1 1.414214 1.732051 2 2.236068
```

- 資料回傳為加總

```
foreach(i = 1:5, .combine='+') %do% sqrt(i)
```

```
[1] 8.382332
```

交易優化的方式

- 參數掃描
 - 類似brute-force解法，掃描參數所有排列組合，從中找到最適切的交易參數
 - 涵蓋了所有的策略參數
 - 易陷入過適(overfitting)的問題。
 - 通常搭配WFA 或蒙地卡羅做策略優化
- MAE/MFE - 圖形方式
- 四分位數(quantiles) - 統計方式
- 前推進式回測(walk forward)
 - 時間遞移的反覆回測下，利用目標函式決定來改變參數的值

參數掃描回測的平行運算應用

- 類似brute-force解法，掃描參數所有排列組合，從中找到最適切的交易參數
- 涵蓋了所有的策略參數
- 易陷入過適(overfitting)的問題。
- 通常搭配WFA 或蒙地卡羅做策略優化

回測平行運算的應用

- `apply.paramset()`
- 策略參數內，產生所有分佈與限制的排列組合
- 每種參數組合各執行 `applyStrategy()` 一次
- `nSamples`: 抽取隨機樣本數目
- 交易帳稽核: 以檔名來分別，儲存每個投資組合與交易帳

apply.paramset範例 - Luxor 雙平均移動線策略

- 原easylanguage 程式碼

```
Inputs: FastLength ( 3 ), SlowLength ( 30 );
```

```
Variables:
```

```
Fast ( 0 ), Slow ( 0 ), GoLong ( F a l s e ), GoShort ( F a l s e ),  
BuyStop ( 0 ), SellStop ( 0 ), BuyLimit ( 0 ), SellLimit ( 0 );
```

```
Fast = Average ( Close , FastLength );
```

```
Slow = Average ( Close , SlowLength );
```

```
GoLong = Fast > Slow ;
```

```
GoShort = Fast < Slow ;
```

```
If Fast crosses above Slow then begin
```

```
BuyStop = High + 1 point ;
```

```
BuyLimit = High + 5 points ;
```

```
end ;
```

```
If Fast crosses below Slow then begin
```

```
SellStop = Low - 1 point ;
```

```
SellLimit = Low - 5 points ;
```

```
end ;
```

```
If GoLong and C < BuyLimit then
```

```
Buy ( " Long " ) next bar at BuyStop Stop ;
```

```
If GoShort and C > SellLimit then
```

```
Sell Short ( " Short " ) next bar at SellStop Stop ;
```

Luxor 雙均線策略假設參數

回測時間:	2002-10-21 ~ 2002-10-31
交易資料解析度	30分鐘 OHLC
交易邏輯:	momentum
	短均線過長均線高買 短均線破長均線低賣
本金:	USD 100,000
交易成本:	USD 6
部位管理:	固定比率
	無加減碼規則
交易商品	外匯 GBPUSD

apply.paramset範例

- 資料載入與時區幣別等設定

```
library(quantstrat)
```

```
#####  
# Luxor 外匯商品交易 雙均線策略 參數設定
```

```
Sys.setenv(TZ="UTC")
```

```
initDate = '2002-10-21'  
  .from=initDate  
  .to='2002-10-31'
```

```
currency(c('GBP', 'USD'))  
exchange_rate('GBPUSD', tick_size=0.0001)
```

```
getSymbols.FI(Symbols='GBPUSD',  
              dir=system.file('extdata',package='quantstrat'),  
              from=.from, to=.to)
```

```
GBPUSD = to.minutes30(GBPUSD)  
GBPUSD = align.time(GBPUSD, 1800)
```

apply.paramset範例

- 雙均線參數相關設定

```
#####  
# 雙均線參數相關設定  
  
# moving average lengths  
.fast = 10  
.slow = 30  
  
# optimization range  
.FastSMA = (1:30)  
.SlowSMA = (20:80)  
  
# trade parameters  
.threshold = 0.0005  
.orderqty = 100000  
.txnfees = -6 # round-trip fee  
  
# stop loss amount  
.stoploss <- 0.30/100  
.StopLoss = seq(0.05, 0.6, length.out=48)/100  
  
# trading window  
.timespan = 'T00:00/T23:59'  
  
# number of optimization samples  
.nsamples=80  
  
portfolio.st = 'forex'  
account.st = 'IB1'  
strategy.st = 'luxor'
```

apply.paramset範例

- 啟始化帳本與投資組合相關資料庫

```
#####
```

```
# 啟始化帳本與投資組合相關資料庫
```

```
rm.strat(portfolio.st)
```

```
rm.strat(account.st)
```

```
initPortf(portfolio.st, symbols='GBPUSD', initDate=initDate, currency='USD')
```

```
initAcct(account.st, portfolios=portfolio.st, initDate=initDate, currency='USD')
```

```
initOrders(portfolio.st, initDate=initDate)
```

```
strategy(strategy.st, store=TRUE)
```

apply.paramset範例

- 新增指標

```
#####  
# 新增指標  
  
add.indicator(strategy.st, name = "SMA",  
               arguments = list(  
                 x = quote(C1(mktdata)[,1]),  
                 n = .fast  
               ),  
               label="nFast"  
)  
  
add.indicator(strategy.st, name="SMA",  
               arguments = list(  
                 x = quote(C1(mktdata)[,1]),  
                 n = .slow  
               ),  
               label="nSlow"  
)
```

apply.paramset範例

- 新增訊號

```
#####  
# 新增訊號  
  
add.signal(strategy.st, name='sigCrossover',  
            arguments = list(  
                columns=c("nFast","nSlow"),  
                relationship="gte"  
            ),  
            label='long'  
)  
  
add.signal(strategy.st, name='sigCrossover',  
            arguments = list(  
                columns=c("nFast","nSlow"),  
                relationship="lt"  
            ),  
            label='short'  
)
```

apply.paramset範例

- 新增規則-進場

```
#####
```

```
# 新增交易規則
```

```
add.rule(strategy.st, name='ruleSignal',
          arguments=list(sigcol='long' , signal=TRUE,
                        orderside='long' ,
                        ordertype='stoplimit',
                        prefer='High',
                        threshold=.threshold,
                        orderqty=+.orderqty,
                        replace=FALSE
          ),
          type='enter',
          label='EnterLONG'
)

add.rule(strategy.st, name='ruleSignal',
          arguments=list(sigcol='short', signal=TRUE,
                        orderside='short',
                        ordertype='stoplimit',
                        prefer='Low',
                        threshold=-.threshold,
                        orderqty=-.orderqty,
                        replace=FALSE
          ),
          type='enter',
          label='EnterSHORT'
)
```

apply.paramset範例

- 新增規則-出場

```
#####  
# 新增交易規則  
  
add.rule(strategy.st, name='ruleSignal',  
          arguments=list(sigcol='short', signal=TRUE,  
                          orderside='long',  
                          ordertype='market',  
                          orderqty='all',  
                          TxnFees=.txnfees,  
                          replace=TRUE  
          ),  
          type='exit',  
          label='Exit2SHORT'  
)  
  
add.rule(strategy.st, name='ruleSignal',  
          arguments=list(sigcol='long', signal=TRUE,  
                          orderside='short',  
                          ordertype='market',  
                          orderqty='all',  
                          TxnFees=.txnfees,  
                          replace=TRUE  
          ),  
          type='exit',  
          label='Exit2LONG'  
)
```

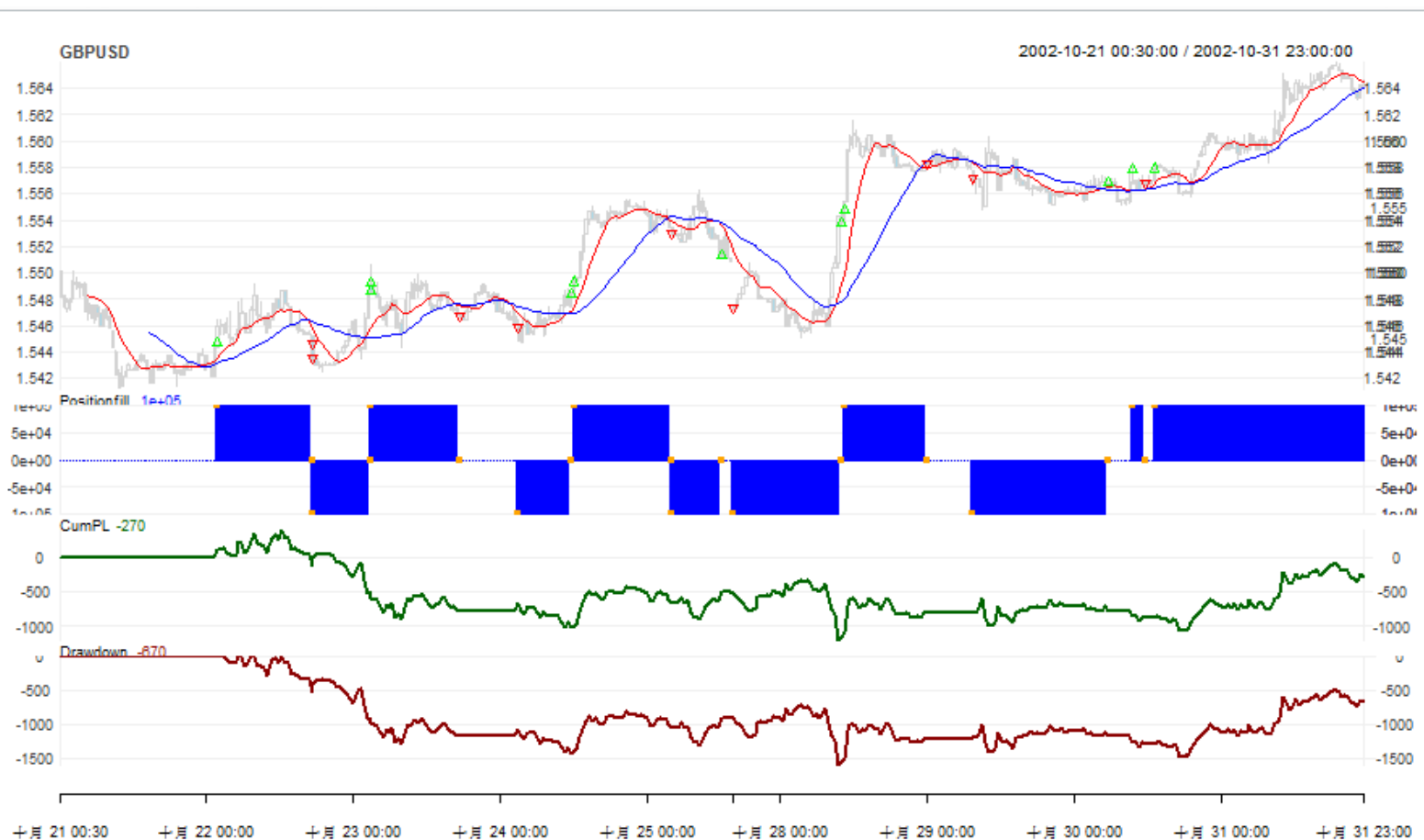
apply.paramset範例

- 執行回測 更新帳戶資料 並呈現回測結果

```
#####  
# 開始執行回測 並更新帳戶資料  
out <- applyStrategy(strategy.st, portfolio.st)  
  
updatePortf(portfolio.st, Symbols='GBPUSD',  
             Dates=paste(':',as.Date(Sys.time())),sep=''))  
  
chart.Posn(portfolio.st, "GBPUSD",  
            TA="add_SMA(n=10,col=2);add_SMA(n=30,col=4)",theme=myTheme)  
  
PerformanceAnalytics:::textplot(t(tradeStats(portfolio.st, 'GBPUSD')))  
  
mk <- mktdata['2002-10-23 15:00::2002-10-24 03:00']  
mk.df <- data.frame(Date=time(mk),coredata(mk))  
PerformanceAnalytics:::textplot(mk.df,show.rownames=F)  
  
ob <- getOrderBook(portfolio.st)$forex$GBPUSD  
ob.df <- data.frame(Date=time(ob),coredata(ob))  
  
PerformanceAnalytics:::textplot(ob.df,show.rownames=F)  
  
PerformanceAnalytics:::textplot(perTradeStats(portfolio.st,"GBPUSD"),  
                                show.rownames=F)
```


apply.paramset範例

- 執行回測 更新帳戶資料 並呈現回測結果



apply.paramset範例

- 設定參數掃描區間

```
#####  
# 設定參數掃描區間  
  
add.distribution(strategy.st,  
                  paramset.label = 'SMA',  
                  component.type = 'indicator',  
                  component.label = 'nFast',  
                  variable = list(n = .FastSMA),  
                  label = 'nFAST'  
)  
  
add.distribution(strategy.st,  
                  paramset.label = 'SMA',  
                  component.type = 'indicator',  
                  component.label = 'nSlow',  
                  variable = list(n = .SlowSMA),  
                  label = 'nSLOW'  
)
```

apply.paramset範例

- 設定參數掃描限制條件

```
#####  
# 設定參數掃描的限制條件  
  
add.distribution.constraint(strategy.st,  
                             paramset.label = 'SMA',  
                             distribution.label.1 = 'nFAST',  
                             distribution.label.2 = 'nSLOW',  
                             operator = '<',  
                             label = 'SMA'  
)
```

apply.paramset範例

- 設定參數掃描限制條件

```
#####
```

```
# 重新啟始化帳本與投資組合相關資料庫
```

```
rm.strat(portfolio.st)
```

```
rm.strat(account.st)
```

```
initPortf(portfolio.st, symbols='GBPUSD', initDate=initDate, currency='USD')
```

```
initAcct(account.st, portfolios=portfolio.st,  
          initDate=initDate, currency='USD')
```

```
initOrders(portfolio.st, initDate=initDate)
```

apply. paramset範例

- 啟動多核心平行運算

```
#####
```

```
# 平行運算起始化
```

```
library(doParallel)
```

```
# Mac/Linux 用doMC
```

```
# library(doMC)
```

```
detectCores()
```

```
if( Sys.info()['sysname'] == "Windows" )
```

```
{
```

```
  library(doParallel)
```

```
  registerDoParallel(cores=detectCores())
```

```
} else {
```

```
  library(doMC)
```

```
  registerDoMC(cores=detectCores())
```

```
}
```

apply.paramset範例

- 開始多核心參數掃描計算

```
#####
```

```
# 用目前CPU所有核心，開始參數掃描計算
```

```
results <- apply.paramset(strategy.st, paramset.label='SMA',  
                           portfolio.st=portfolio.st, account.st=account.st,  
                           nsamples=0)
```

```
#####
```

```
# 回測結果用常短天參數順序排列
```

```
tS <- results$tradeStats  
idx <- order(tS[,1],tS[,2])  
tS <- tS[idx,]  
PerformanceAnalytics:::textplot(t(tS)[,1:10])
```

apply.paramset範例

- 參數從大到小排列組合結果

	1	20	40	61	83	106	130	155	181	208
nFAST	1	1	1	1	1	1	1	1	1	1
nSLOW	20	21	22	23	24	25	26	27	28	29
Portfolio	forex.1	forex.20	forex.40	forex.61	forex.83	forex.106	forex.130	forex.155	forex.181	forex.208
Symbol	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD	GBPUSD
Num.Txns	55	59	57	53	53	53	47	47	46	40
Num.Trades	27	29	28	26	26	26	23	23	23	20
Net.Trading.PL	-372.000000	-1214.000000	-808.000000	-716.000000	-796.000000	-1136.000000	-918.000000	-688.000000	-418.000000	-280.000000
Avg.Trade.PL	-11.55555556	-39.79310345	-26.71428571	-25.23076923	-28.30769231	-41.38461538	-37.30434783	-27.30434783	-18.17391304	-14.00000000
Med.Trade.PL	-86.0000000	-116.0000000	-106.0000000	-96.0000000	-96.0000000	-96.0000000	-96.0000000	-96.0000000	-96.0000000	-96.0000000
Largest.Winner	944.000000	944.000000	944.000000	954.000000	944.000000	674.000000	674.000000	674.000000	694.000000	674.000000
Largest.Loser	-336	-506	-506	-506	-506	-506	-506	-506	-506	-406
Gross.Profits	2282.000000	2208.000000	2208.000000	2134.000000	2054.000000	1714.000000	1754.000000	1984.000000	2008.000000	2040.000000
Gross.Losses	-2594.0000	-3362.0000	-2956.0000	-2790.0000	-2790.0000	-2790.0000	-2612.0000	-2612.0000	-2426.0000	-2320.0000
Std.Dev.Trade.PL	261.08256	268.25379	266.72122	280.69803	273.90959	234.96776	249.75086	259.25250	257.85755	284.94875
Percent.Positive	29.6296296	24.1379310	25.0000000	23.0769231	23.0769231	23.0769231	26.0869565	26.0869565	30.4347826	25.0000000
Percent.Negative	70.370370	75.862069	75.0000000	76.923077	76.923077	76.923077	73.913043	73.913043	69.565217	75.0000000
Profit.Factor	0.8797224364	0.6567519334	0.7469553451	0.7648745520	0.7362007168	0.6143369176	0.6715160796	0.7595712098	0.8276999176	0.8793103448
Avg.Win.Trade	285.250000	315.428571	315.428571	355.666667	342.333333	285.666667	292.333333	330.666667	286.857143	408.000000
Med.Win.Trade	164.000000	194.000000	194.000000	199.000000	199.000000	164.000000	184.000000	264.000000	214.000000	524.000000
Avg.Losing.Trade	-136.52632	-152.81818	-140.76190	-139.50000	-139.50000	-139.50000	-153.64706	-153.64706	-151.62500	-154.66667
Med.Losing.Trade	-136.000000	-136.000000	-136.000000	-131.000000	-131.000000	-131.000000	-146.000000	-146.000000	-121.000000	-176.000000
Avg.Daily.PL	-34.66666667	-128.22222222	-83.11111111	-72.88888889	-81.77777778	-119.55555556	-107.25000000	-78.50000000	-52.25000000	-35.00000000
Med.Daily.PL	-8.60000000e+01	-8.60000000e+01	-8.40000000e+01	-1.60000000e+01	-1.60000000e+01	-1.60000000e+01	-1.12000000e+02	-1.12000000e+02	-3.70000000e+01	-9.20000000e+01
Std.Dev.Daily.PL	524.83331	677.74143	606.37786	615.15048	621.84439	536.84614	601.95295	625.81764	601.09561	540.68317
Ann.Sharpe	-1.0485543991	-3.0033056687	-2.1757852167	-1.8809629198	-2.0876315668	-3.5352505431	-2.8283621856	-1.9912332197	-1.3798853736	-1.0276032444
Max.Drawdown	-1118.0000	-1960.0000	-1624.0000	-1558.0000	-1604.0000	-1604.0000	-1870.0000	-1870.0000	-1684.0000	-1568.0000
Profit.To.Max.Draw	-0.3327370304	-0.6193877551	-0.4975369458	-0.4595635430	-0.4962593516	-0.7082294264	-0.4909090909	-0.3679144385	-0.2482185273	-0.1785714286
Avg.WinLoss.Ratio	2.0893407864	2.0640775049	2.2408660352	2.5495818399	2.4540023895	2.0477897252	1.9026288923	2.1521184278	1.8918855259	2.6379310345
Med.WinLoss.Ratio	1.2058823529	1.4264705882	1.4264705882	1.5190839695	1.5190839695	1.2519083969	1.2602739726	1.8082191781	1.7685950413	2.9772727273
Max.Equity	714.000000	198.000000	278.000000	440.000000	360.000000	308.000000	324.000000	324.000000	324.000000	314.000000
Min.Equity	-920.000000	-1762.000000	-1356.000000	-1250.000000	-1296.000000	-1296.000000	-1546.000000	-1546.000000	-1360.000000	-1254.000000
End.Equity	-372.000000	-1214.000000	-808.000000	-716.000000	-796.000000	-1136.000000	-918.000000	-688.000000	-418.000000	-280.000000

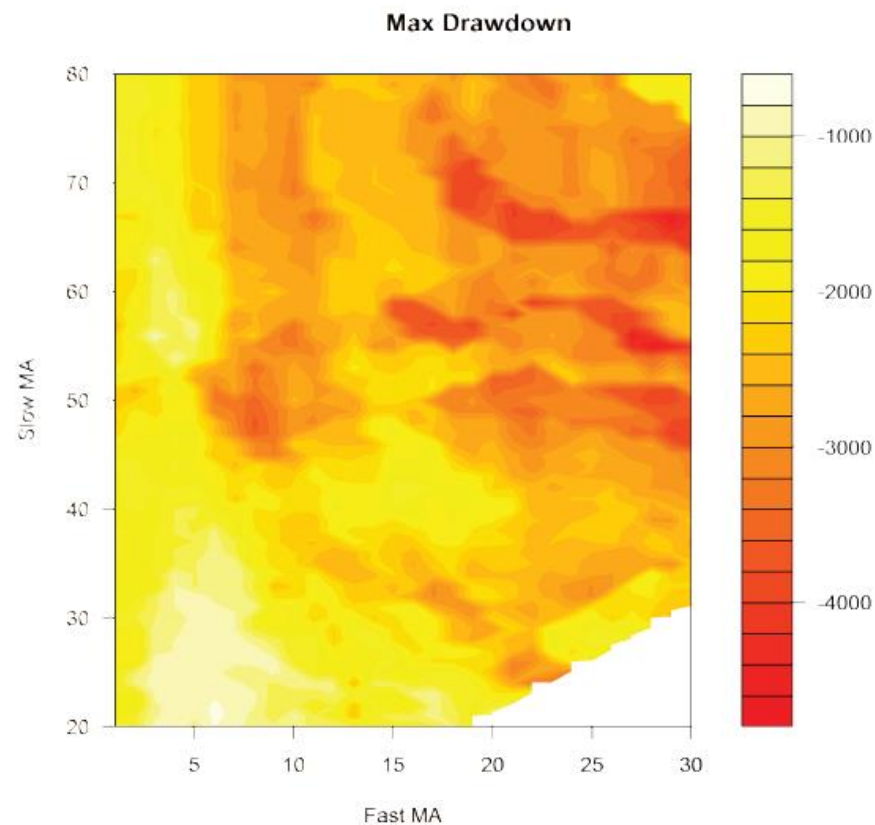
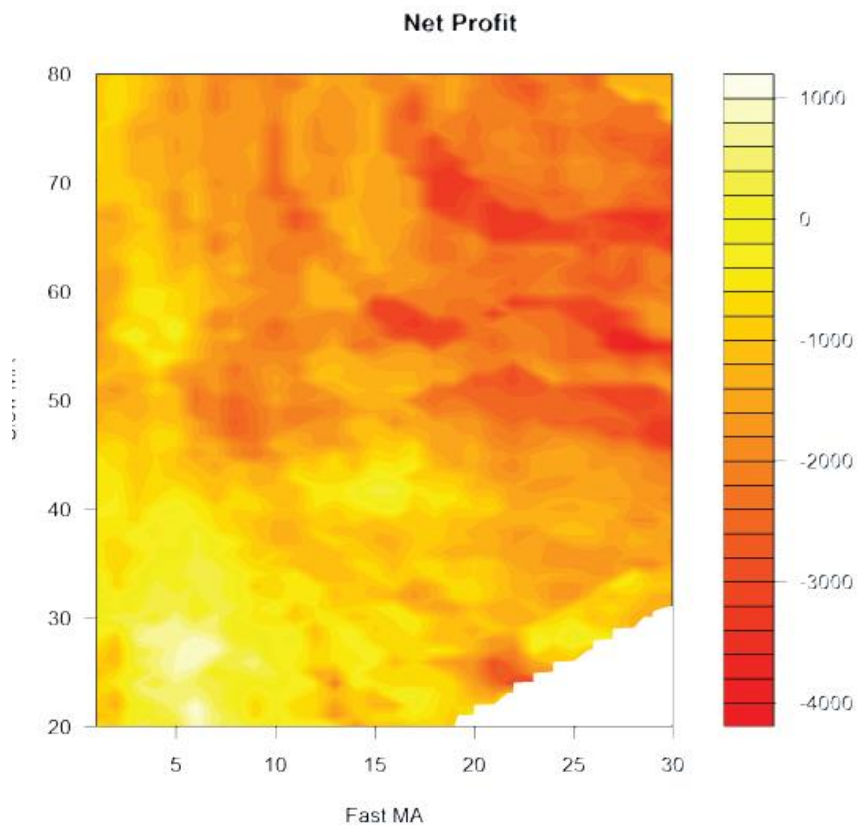
apply. paramset範例

- 收益與風險 熱區圖

```
#####  
# 收益與風險 熱區圖  
# net profit  
z <- tapply(X=tS[, "End.Equity"], INDEX=list(Fast=tS[,1], Slow=tS[,2]), FUN=sum)  
  
x <- as.numeric(rownames(z))  
y <- as.numeric(colnames(z))  
filled.contour(x=x, y=y, z=z, color = heat.colors, xlab="Fast MA", ylab="Slow  
MA")  
title("淨利(NP)")  
  
# MaxDD  
z <-  
tapply(X=tS[, "Max.Drawdown"], INDEX=list(Fast=tS[,1], Slow=tS[,2]), FUN=sum)  
x <- as.numeric(rownames(z))  
y <- as.numeric(colnames(z))  
filled.contour(x=x, y=y, z=z, color = heat.colors, xlab="Fast MA", ylab="Slow  
MA")  
title("最大回檔(MDD)")
```


apply.paramset 範例

- 收益與風險 熱區圖



apply. paramset範例

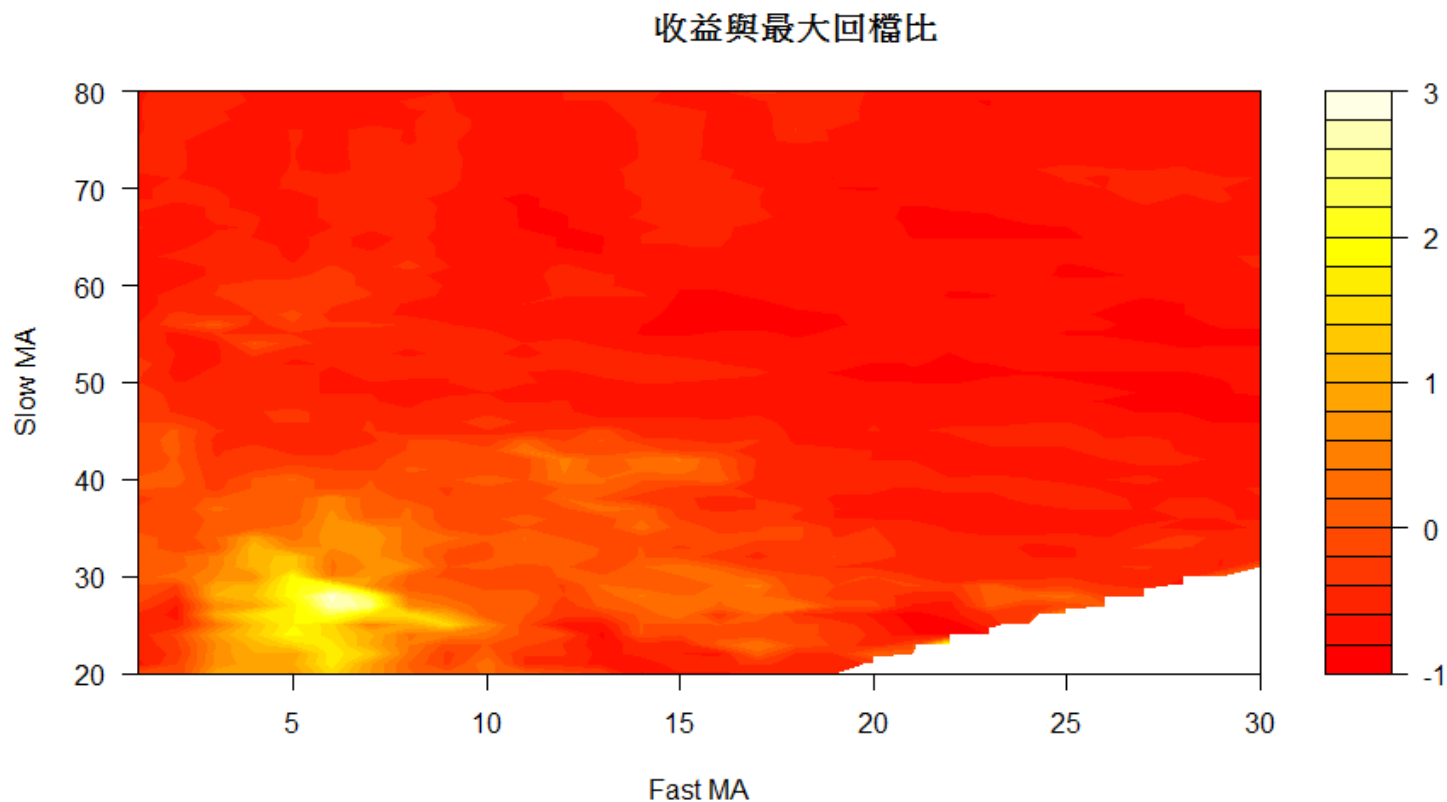
- 收益與最大回檔比 熱區圖

```
# return to maxdd
z <- tapply(X=tS[, "Profit.To.Max.Draw"],
            INDEX=list(Fast=tS[,1], Slow=tS[,2]), FUN=sum)
x <- as.numeric(rownames(z))
y <- as.numeric(colnames(z))
filled.contour(x=x, y=y, z=z, color = heat.colors, xlab="Fast MA", ylab="Slow
MA")
title("收益與最大回檔比")
```

```
rmdd <- tS$Profit.To.Max.Draw
idx <- order(rmdd, decreasing=T)[1:30]
labs <- paste(tS$nFAST[idx], tS$nSLOW[idx], sep="/")
barplot(rmdd[idx], names.arg=labs, col=4, las=2, main="收益與最大回檔比值")
```

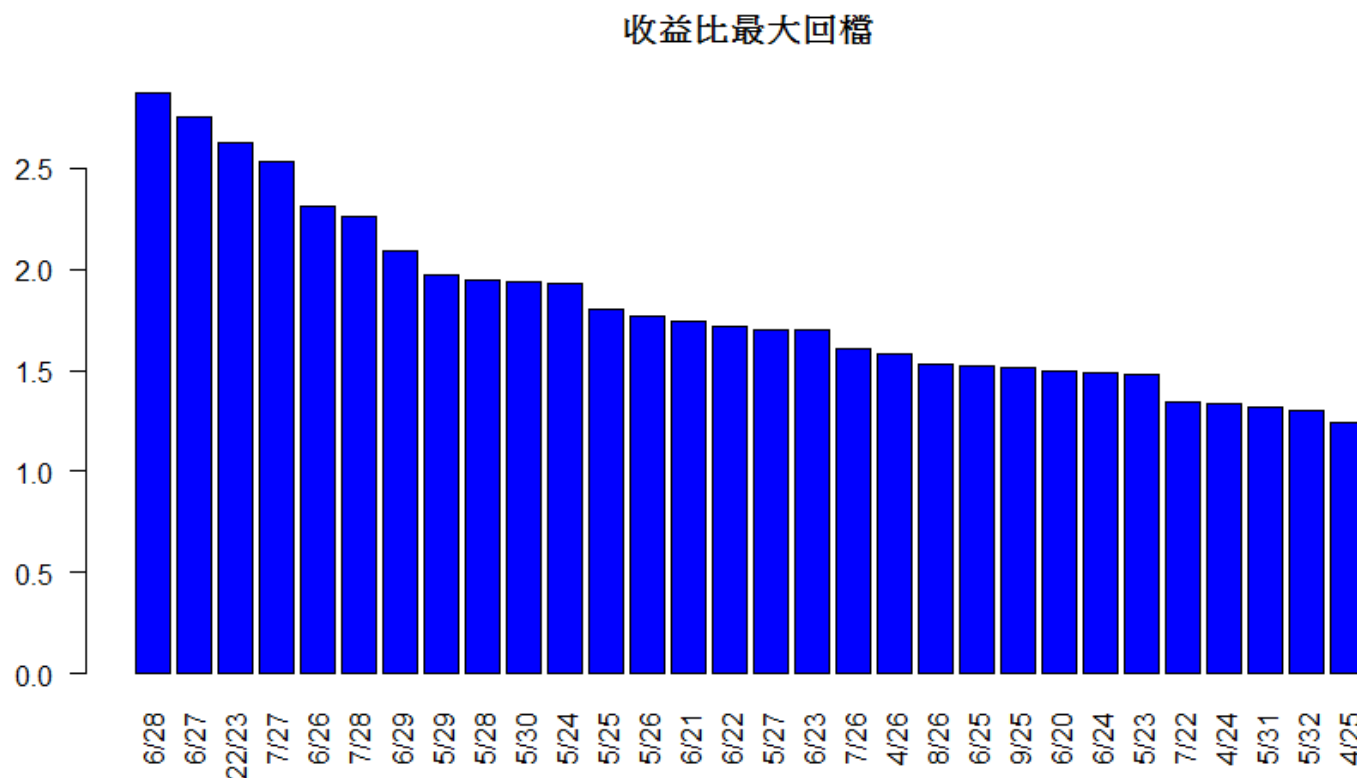
apply.paramset 範例

- 收益與最大回檔比 熱區圖



apply.paramset 範例

- 收益與最大回檔比 柱狀圖



參數高原3D圖

- 安裝rgl 套件 (R的OpenGL 3D圖形函式庫)

```
install.packages("rgl")
```

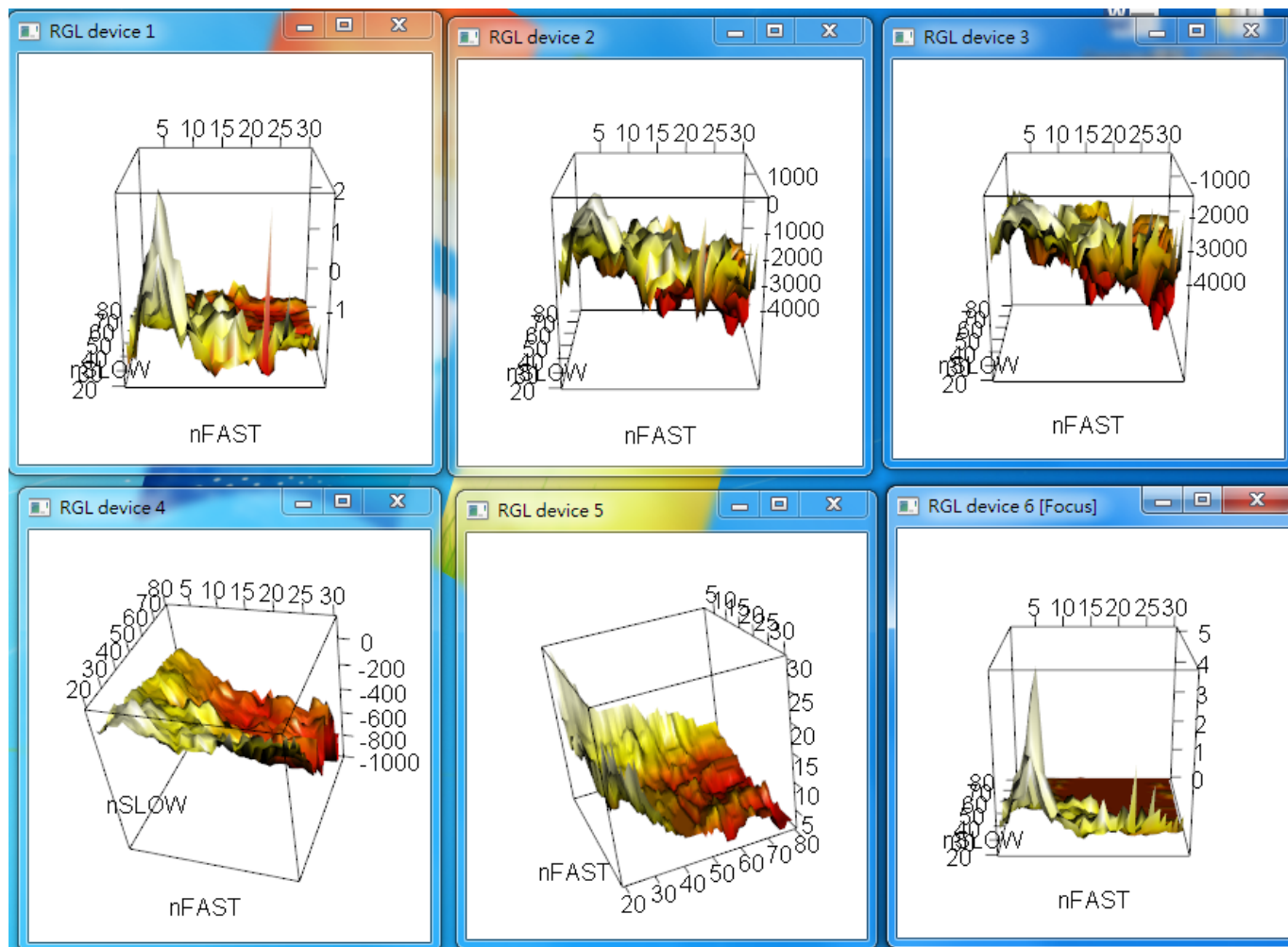
```
#####
```

```
# 參數高原3D 圖
```

```
# net profit
```

```
tradeGraphs (stats = tS, free.params = c("nFAST", "nSLOW"),  
              statistics = c("Profit.To.Max.Draw", "Net.Trading.PL",  
                             "Max.Drawdown", "Avg.Trade.PL", "Num.Trades", "Profit.Factor"),  
              title = '')
```

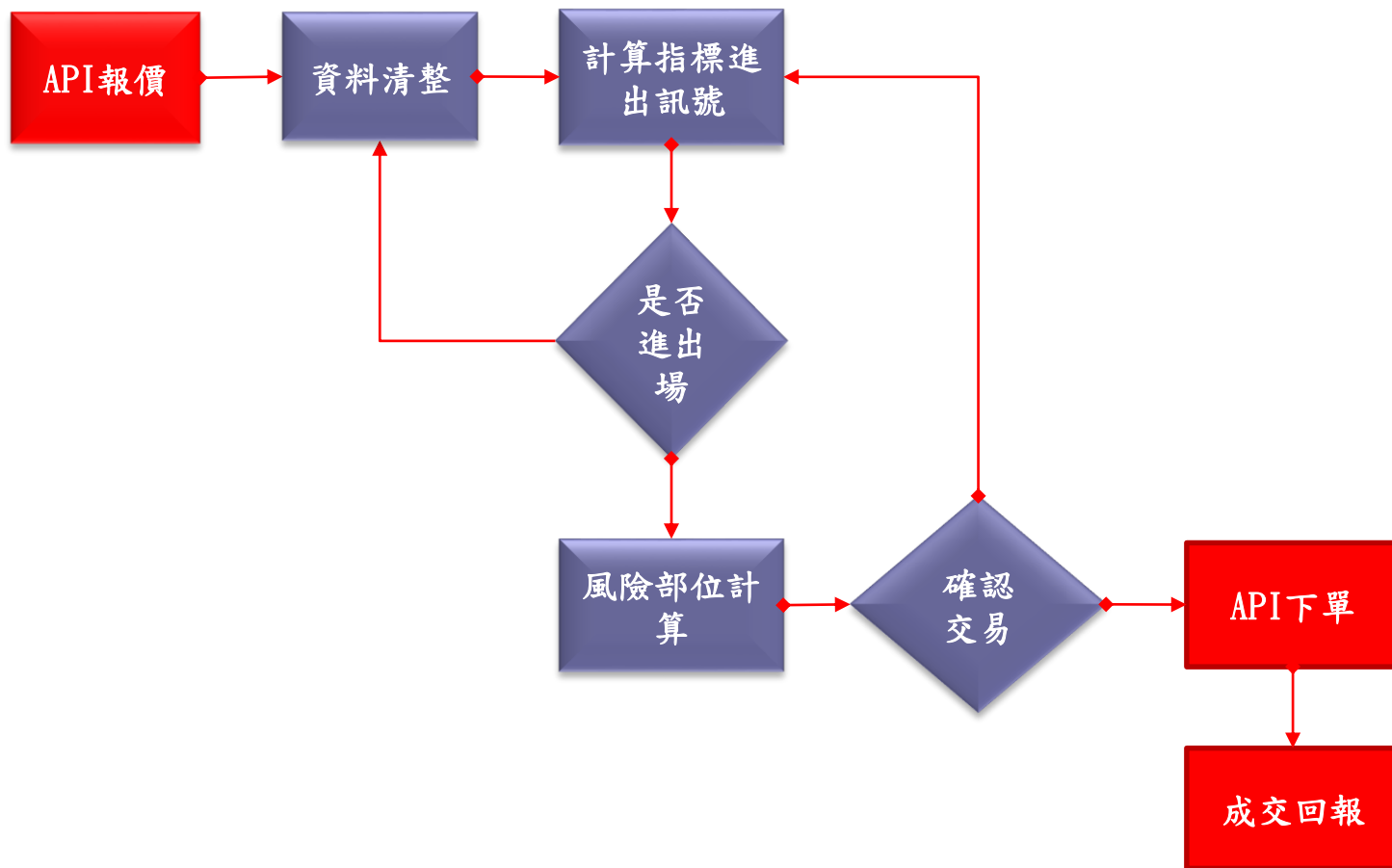
參數高原3D圖



如何應用R套件於實際交易

API 下單程式流程

- 下單流程



歷史資料準備

- 設定套件與檔案

```
library(xts)
library(timeDate)
library(lubridate)
library(quantmod)
require(doParallel)

# 下載檔放置位置
hist_dir <- 'C:/Users/julian/Downloads/hist/test'
setwd(hist_dir)

# 使用平行運算，加快資料重整速度
cl <- makeCluster(4)
registerDoParallel(cl)
```

歷史資料清整與型態互轉(coercion)

- 資料載入階段

指數期貨資料轉換主程式

```
TXFExtract <- function(qryyear, qrymonth, qryday){
```

```
  # 判斷當日屬於哪一近月結算月
```

```
  settleMonth <- timeNthNdayInMonth(paste(qryyear, qrymonth, '01', sep = '-'), 3, 2)
```

```
  if(as.Date(paste(qryyear, qrymonth, qryday, sep = '-')) >  
as.Date(settleMonth["Data"])){
```

```
    print("leap")
```

```
    curStlMth <- substr(as.character(as.Date(paste(qryyear, qrymonth, '01', sep = '-'))  
%m+% months(c(1))), 1, 7)
```

```
    curStlMth <- gsub('-', '', curStlMth)
```

```
  }else{
```

```
    print("not leap")
```

```
    curStlMth <- paste(qryyear, qrymonth, sep = '-')
```

```
  }
```

```
  print(curStlMth)
```

```
  unzip(zipfile=paste(hist_dir, "/Daily_", qryyear, "_", qrymonth, "_", qryday,  
".zip", sep = ''),  
    exdir=".")
```

```
  tx <- read.csv(paste(hist_dir, "/Daily_", qryyear, "_", qrymonth, "_", qryday,  
".csv", sep = ''),  
    header = FALSE, skip = 1, sep = ",")
```

```
  unlink(paste("Daily_", qryyear, "_", qrymonth, "_", qryday, ".csv", sep = ''))
```

歷史資料清整與型態互轉(coercion)

- 資料清整階段

```
# 清整資料
# Trim spaces
tx$V2 =trimws(tx$V2)
tx$V3 =trimws(tx$V3)

tx <- tx[(tx$V3 == curStlMth & tx$V2 == "TX"),]

tx <- subset(tx, select = c("V1", "V4", "V5", "V6"))

colnames(tx) <- c("Date", "Time", "Price", "Volume")
tx$Date <- as.character(tx$Date)
tx$Time <- as.character(tx$Time)
tx$Price <- as.integer(tx$Price)
tx$Volume <- as.integer(tx$Volume)/2

# 填齊8位時間個數
tx[nchar(tx$Time) == 5,]$Time <- paste('0', tx[nchar(tx$Time) == 5,]$Time,
sep = '')

tx$Date <- paste(substr(tx$Date, 1, 4), '-', substr(tx$Date, 5, 6), '-',
substr(tx$Date,7, 8),sep = '')
tx$Time <- paste(substr(tx$Time, 1, 2), ':', substr(tx$Time, 3, 4),':',
substr(tx$Time,5, 6),sep = '')
tx$DateTime <- paste(tx$Date, tx$Time )

tx <- subset(tx, select = c(DateTime, Price, Volume))
```

歷史資料清整與型態互轉(coercion)

- 時間序列物件階段

```
# Data Frame 轉型為xts 時間序列物件
tx_xts <- xts(tx[, -1], order.by=as.POSIXct(tx$DateTime))

# 轉換tick 為分鐘 OHLC Bar data
bars_1min <- period.apply(tx_xts,
                          endpoints(tx_xts,"secs",60),
                          function(xx){
                            ticks=coredata(xx$Price)
                            c( first(ticks),max(ticks), min(ticks),
                               last(ticks), sum(xx$Volume) )
                          })
colnames(bars_1min) <- c("Open","High","Low","Close","Volume")
align.time(bars_1min,60)

return(bars_1min)
}
```

轉換主程式

- 批次把整個目錄下的歷史資料轉換成可以回測與實際交易的資料庫

```
# 掃描目錄，取Daily開頭的zip壓縮檔檔名
file.ls <- list.files(path=hist_dir, pattern=glob2rx("Daily_*.zip"))

for(l in seq_along(file.ls)){

  qryyear <- substr(file.ls[[l]], 7, 10)
  qrymonth <- substr(file.ls[[l]], 12, 13)
  qryday <- substr(file.ls[[l]], 15, 16)

  if(l == 1 ){
    txflmin <- TXFExtract(qryyear, qrymonth, qryday)
  }else{
    txflmin <- rbind(txflmin, TXFExtract(qryyear, qrymonth, qryday))
  }

  print(sprintf("共 %s 個交易日歷史資料，目前轉第 %s 個檔，日期： %s-%s-%s ",
length(file.ls), l, qryyear, qrymonth, qryday ))

}
```

資料讀入與計算

- 批次轉檔畫面

```
Console C:/Users/julian/Downloads/hist/test/
> source('C:/Users/julian/Desktop/證基會第四季課程/Day 3/程式碼 資料/practices/tx_backtest.R', encoding = 'UTF-8')
[1] "not leap"
[1] "201607"
[1] "共 62 個交易日歷史資料，目前轉第 1 個檔，日期：2016-07-01 "
[1] "not leap"
[1] "201607"
[1] "共 62 個交易日歷史資料，目前轉第 2 個檔，日期：2016-07-04 "
[1] "not leap"
[1] "201607"
[1] "共 62 個交易日歷史資料，目前轉第 3 個檔，日期：2016-07-05 "
[1] "not leap"
[1] "201607"
[1] "共 62 個交易日歷史資料，目前轉第 4 個檔，日期：2016-07-06 "
[1] "not leap"
[1] "201607"
[1] "共 62 個交易日歷史資料，目前轉第 5 個檔，日期：2016-07-07 "
```

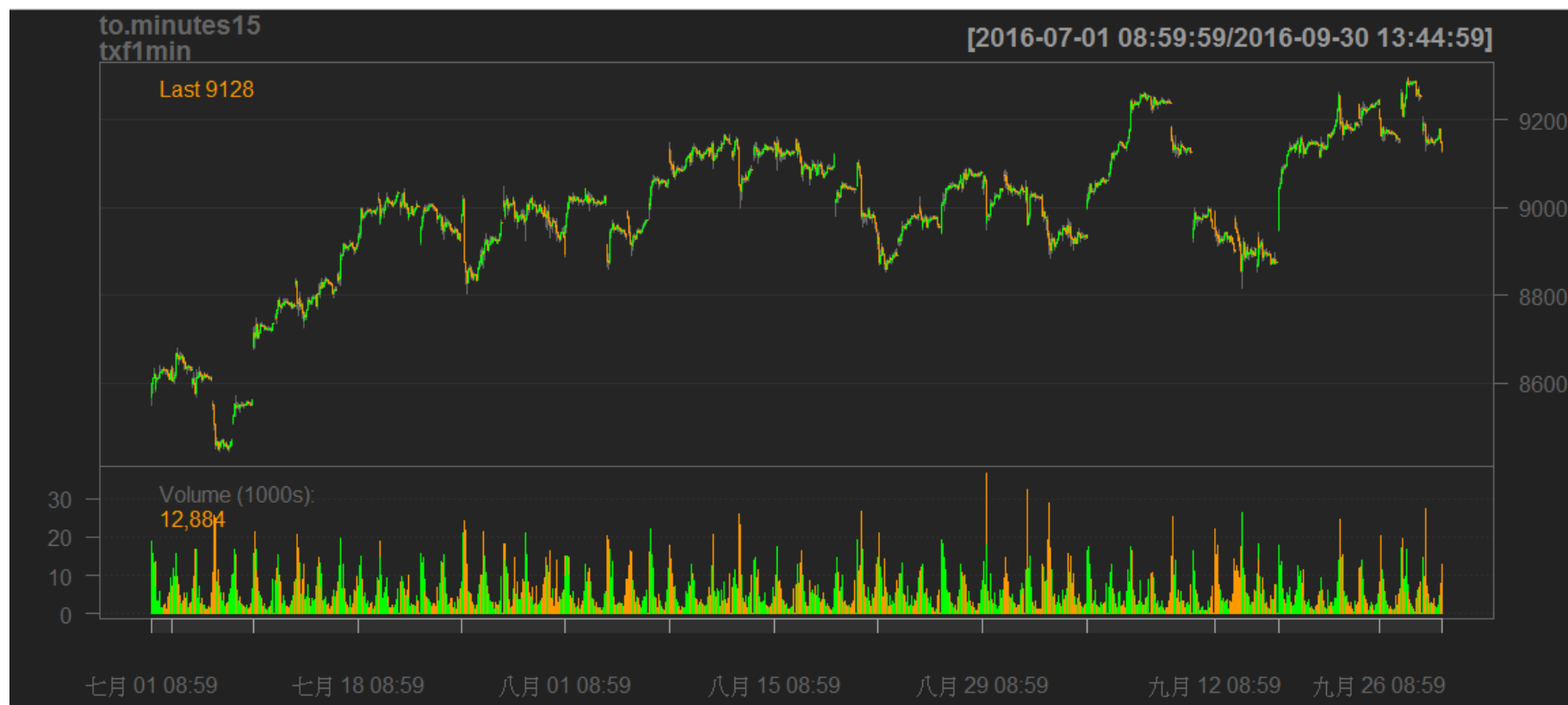
- 完成後產生1分鐘開高收低歷史資料檔

Global Environment ▾		🔍
Data		
txf1min	Large xts (92950 elements, 873.9 kb)	📅
values		
...		

台指期K線圖表

- 15分鐘K線圖

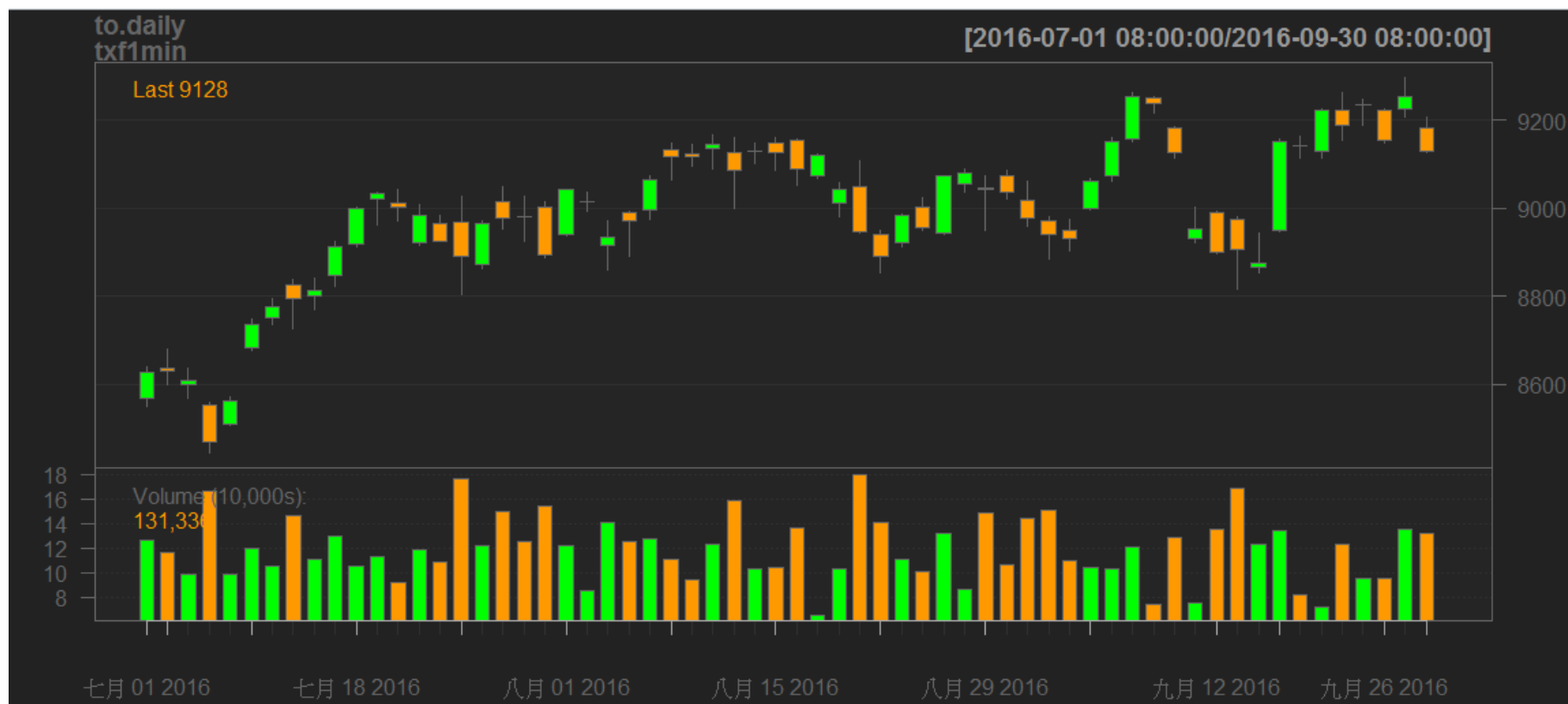
`chartSeries(to.minutes15(txf1min))`



台指期K線圖表

- 日K線圖

`chartSeries(to.daily(txf1min))`



產生進出場訊號

```
# 產生訊號資料表
txf15min <- to.minutes15(txf1min)
txf15min$ssma <- SMA(Cl(txf15min), n=10)
txf15min$slma <- SMA(Cl(txf15min), n=60)

txf15min$up <- NA; txf15min$dn <- NA;
txf15min$up[txf15min$ssma > txf15min$slma, ] <- 1
txf15min$dn[txf15min$ssma < txf15min$slma, ] <- -1

txf15min_sig <- subset(txf15min, select = c(ssma, slma, up, dn))
```

	ssma	slma	up	dn
2016-07-07 11:29:50	8547.8	8555.233	NA	-1
2016-07-07 11:44:57	8547.5	8553.433	NA	-1
2016-07-07 11:59:56	8547.9	8551.850	NA	-1
2016-07-07 12:14:59	8549.2	8550.567	NA	-1
2016-07-07 12:29:59	8549.9	8549.183	1	NA
2016-07-07 12:44:58	8550.6	8547.850	1	NA
2016-07-07 12:59:51	8551.6	8546.467	1	NA
2016-07-07 13:14:58	8552.0	8545.050	1	NA
2016-07-07 13:29:59	8552.3	8543.800	1	NA
2016-07-07 13:44:59	8553.7	8542.667	1	NA
2016-07-11 08:59:59	8570.2	8544.450	1	NA
2016-07-11 09:14:59	8585.9	8546.250	1	NA
2016-07-11 09:29:59	8604.1	8548.833	1	NA

決定進出場規則

- 與下單程式銜接點

```
# 最後一筆 為最近時間的觸動訊號，接下單機制(交易執行程式)  
tail(txf15min_sig, 1)
```

```
if(tail(txf15min_sig, 1)$dn == -1)  
{  
  print("接交易程式下賣出單")  
}
```

```
if(tail(txf15min_sig, 1)$up == 1)  
{  
  print("接交易程式下買入單")  
}
```

API 下單程式說明



課後討論