COMPUTATIONAL FINANCE & RISK MANAGEMENT

UNIVERSITY *of* WASHINGTON

Department of Applied Mathematics

Introduction to Trading Systems

Guy Yollin

Applied Mathematics
University of Washington

Outline

Lecture references

E. Tomasini and U. Jaekle.
Trading Systems: A New Approach to System Development and Portfolio Optimisation.
Harriman House, 2009.

- Chapter 6 - Periodic re-optimization and walk forward analysis

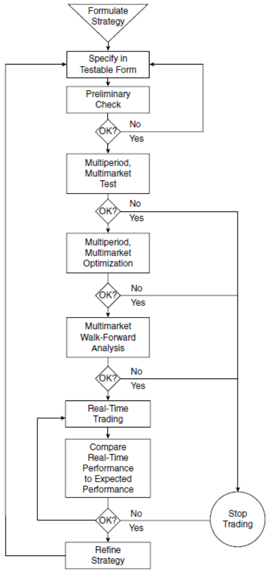
- TradeAnalytics project page on R-forge:
  `http://r-forge.r-project.org/projects/blotter/`
  - documents and demos for:
    - quantstrat package
      (specifically the `Luxor` demo scripts)†

- Using quantstrat by Jan Humme & Brian Peterson
  `http://www.rinfinance.com/agenda/2013/workshop/Humme+Peterson.pdf`

†demos are located in the directory: `.../R-3.x.x/library/quantstrat/demo`

Outline

Trading system development process

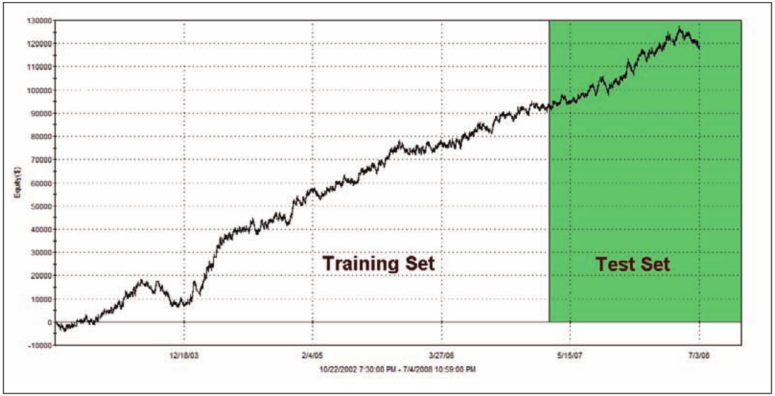


Formulate Strategy
Specify in Testable Form
Preliminary Check
OK?
No
Yes
Multiperiod, Multimarket Test
OK?
No
Yes
Multiperiod, Multimarket Optimization
OK?
No
Yes
Multimarket Walk-Forward Analysis
OK?
No
Yes
Real-Time Trading
Compare Real-Time Performance to Expected Performance
OK?
No
Yes
Refine Strategy
Stop Trading

Evaluation and Optimization of Trading Strategies, R. Pardo

Traditional parameter optimization

- In-sample training window used to develop strategy and optimize strategy parameters
- Out-of-sample testing window used to evaluate system performance

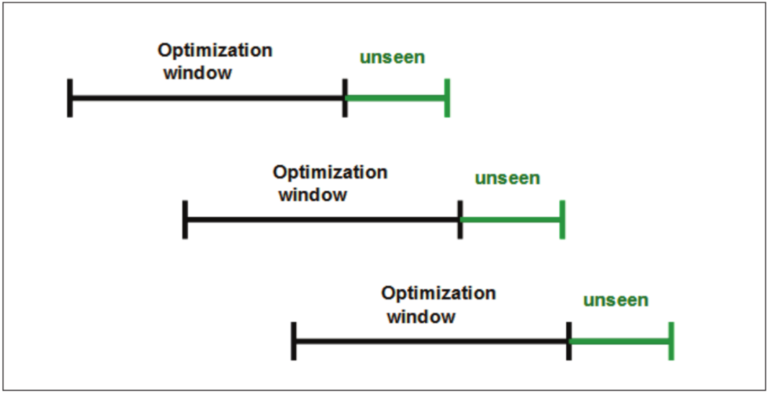


Walk forward analysis

- Walk-forward-analysis involves periodic re-optimization followed by out-of-sample testing
  - Rolling training window (fixed length)
  - Anchored training window (fixed starting point)
- Efficient use of finite sample data
- Allows parameters to adapt to changing market regimes

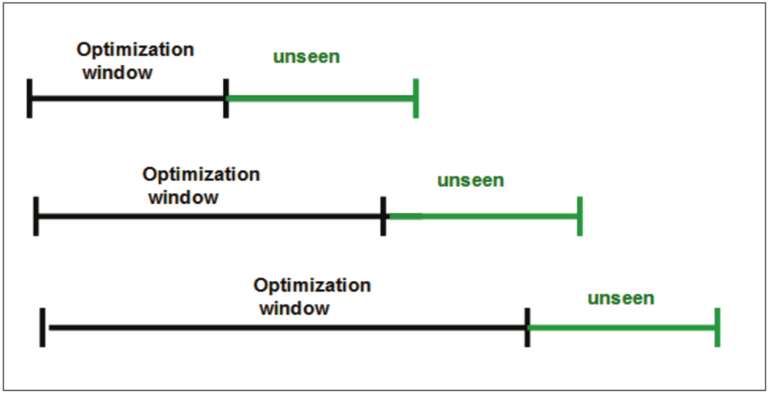
Walk forward analysis

- Training window with a fixed length (rolling window)

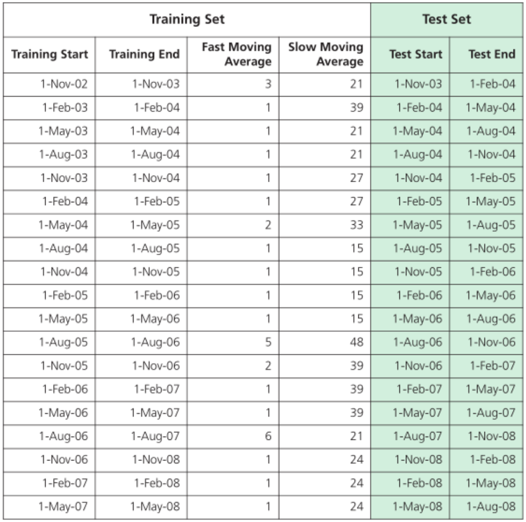


Walk forward analysis

- Traing window with a fixed starting point (anchored window)

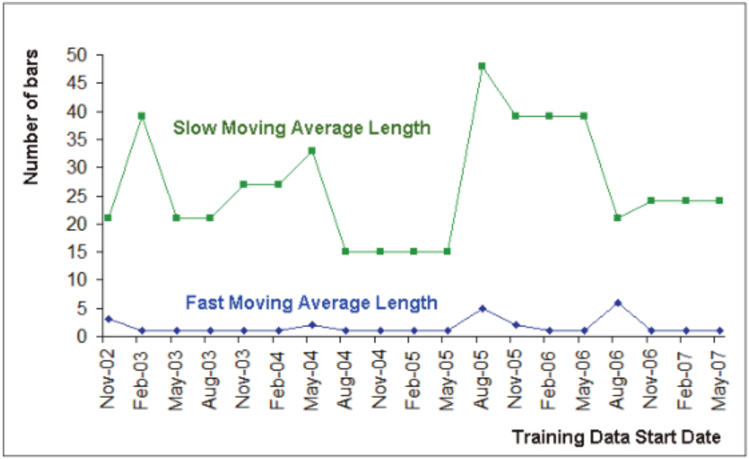


Luxor strategy walk forward analsysis



Training Set				Test Set	
Training Start	Training End	Fast Moving Average	Slow Moving Average	Test Start	Test End
1-Nov-02	1-Nov-03	3	21	1-Nov-03	1-Feb-04
1-Feb-03	1-Feb-04	1	39	1-Feb-04	1-May-04
1-May-03	1-May-04	1	21	1-May-04	1-Aug-04
1-Aug-03	1-Aug-04	1	21	1-Aug-04	1-Nov-04
1-Nov-03	1-Nov-04	1	27	1-Nov-04	1-Feb-05
1-Feb-04	1-Feb-05	1	27	1-Feb-05	1-May-05
1-May-04	1-May-05	2	33	1-May-05	1-Aug-05
1-Aug-04	1-Aug-05	1	15	1-Aug-05	1-Nov-05
1-Nov-04	1-Nov-05	1	15	1-Nov-05	1-Feb-06
1-Feb-05	1-Feb-06	1	15	1-Feb-06	1-May-06
1-May-05	1-May-06	1	15	1-May-06	1-Aug-06
1-Aug-05	1-Aug-06	5	48	1-Aug-06	1-Nov-06
1-Nov-05	1-Nov-06	2	39	1-Nov-06	1-Feb-07
1-Feb-06	1-Feb-07	1	39	1-Feb-07	1-May-07
1-May-06	1-May-07	1	39	1-May-07	1-Aug-07
1-Aug-06	1-Aug-07	6	21	1-Aug-07	1-Nov-08
1-Nov-06	1-Nov-08	1	24	1-Nov-08	1-Feb-08
1-Feb-07	1-Feb-08	1	24	1-Feb-08	1-May-08
1-May-07	1-May-08	1	24	1-May-08	1-Aug-08

Luxor strategy walk forward analsysis



Outline

Load libraries and download data

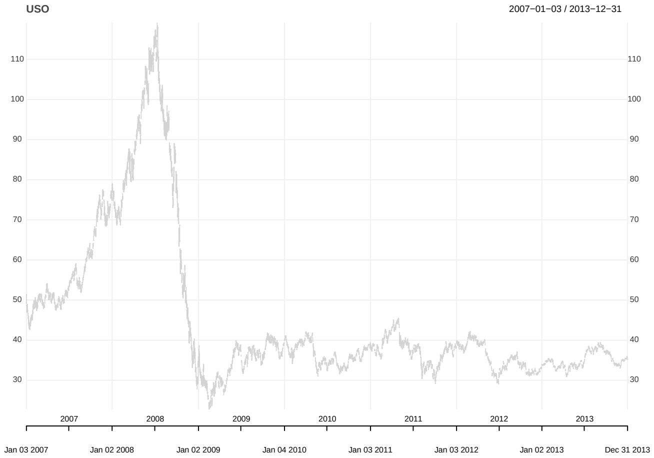
```
library(quantstrat)
library(xtsExtra)
```

```
stock.st = c("USO")

currency("USD")
stock(stock.st, currency="USD",multiplier=1)
Sys.setenv(TZ="UTC")       # set time zone

initDate = '2006-12-31'
startDate = '2007-01-01'
endDate = '2013-12-31'
initEq=1e6
tradeSize = initEq/10

getSymbols(stock.st,from=startDate,to=endDate,index.class="POSIXct",adjust=T)
```

Plot price history

```r
myTheme<-chart_theme()
myTheme$col$dn.col <-'lightblue'
myTheme$col$dn.border <- 'lightgray'
myTheme$col$up.border <- 'lightgray'

chart_Series(get(stock.st),name=stock.st,theme=myTheme)
```

U.S. Oil 2007 to 2013



Define order sizing function

```r
osFixedDollar <- function(timestamp, orderqty, portfolio, symbol, ruletype, ...)
{
  pos <- getPosQty(portfolio, symbol, timestamp)
  if( isTRUE(all.equal(pos,0)) )
  {
    ClosePrice <- as.numeric(Cl(mktdata[timestamp,]))
    orderqty <- sign(orderqty)*round(tradeSize/ClosePrice,-2)
  } else {
    orderqty <- 0
  }
  return(orderqty)
}
```

lnz strategy object, define indicators and signals

```r
strat.st <- "bbands"

rm.strat(strat.st)

strategy(strat.st, store=TRUE)

add.indicator(strat.st, name = "BBands",
  arguments = list(HLC = quote(HLC(mktdata)), maType='SMA'), label='BBands')

add.signal(strat.st, name="sigCrossover",
  arguments=list(columns=c("Close","up"),relationship="gt"),
  label="Cl.gt.UpperBand")

add.signal(strat.st, name="sigCrossover",
  arguments=list(columns=c("Close","dn"),relationship="lt"),
  label="Cl.lt.LowerBand")

add.signal(strat.st, name="sigCrossover",
  arguments=list(columns=c("High","Low","mavg"),relationship="op"),
  label="Cross.Mid")
```

Define rules

```r
add.rule(strat.st, name='ruleSignal',
  arguments=list(sigcol="Cl.gt.UpperBand",sigval=TRUE, orderqty=-100,
    ordertype='market', orderside=NULL, threshold=NULL, osFUN=osFixedDollar,
    orderset='ocoshort'),
        type='enter',label="SE")

add.rule(strat.st, name='ruleSignal',
  arguments=list(sigcol="Cl.lt.LowerBand",sigval=TRUE, orderqty= 100,
    ordertype='market', orderside=NULL, threshold=NULL, osFUN=osFixedDollar,
    orderset='ocolong'),
  type='enter',label="LE")

add.rule(strat.st, name='ruleSignal',
  arguments=list(sigcol="Cross.Mid",sigval=TRUE, orderqty= 'all',
    ordertype='market', orderside=NULL, threshold=NULL),
  type='exit')
```

Define distributions

```r
add.distribution(strat.st,
  paramset.label = 'BBOPT',
  component.type = 'indicator',
  component.label = 'BBands',
  variable = list(n = seq(10,30,by=5)),
  label = 'n'
)

add.distribution(strat.st,
  paramset.label = 'BBOPT',
  component.type = 'indicator',
  component.label = 'BBands',
  variable = list(sd = seq(1,3,by=0.5)),
  label = 'sd'
)
```

Outline

The walk.forward function

The `walk.forward` function is a wrapper for apply.paramset() and applyStrategy(), implementing a Rolling Walk Forward Analysis (WFA).

Usage:

```
args(walk.forward)


## function (strategy.st, paramset.label, portfolio.st, account.st,
##     period, k.training, nsamples = 0, audit.prefix = NULL, k.testing,
##     obj.func = function(x) {
##         which(x == max(x))
##     }, obj.args = list(x = quote(tradeStats.list$Net.Trading.PL)),
##     anchored = FALSE, include.insamples = TRUE, ..., verbose = FALSE)
## NULL
```

Main arguments:

strategy.st	name of the strategy object
paramset.label	unique identifyer of the paramset to be tested
portfolio.st	name of the portfolio object
account.st	name of the account object

The `walk.forward` function

Main arguments (continued):

period	period unit as a string, eg. 'days' or 'months'
k.training	number of training periods, eg. '3' months
k.testing	number of test periods, eg. '1 month'
nsamples	number of sample param.combos to draw; 0 means all samples
obj.func	a user provided function returning the best param.combo
obj.args	a user provided argument to obj.func
anchored	whether to use a fixed start for the training window
include.insamples	will run a full backtest for each param.combo in the paramset
audit.prefix	prefix to generate filenames for storage of audit data
verbose	dumps a lot of info during the run if set to TRUE
...	optional parameters to pass to apply.paramset()

Configure parallel processing

```
if( Sys.info()['sysname'] == "Windows" )
{
  library(doParallel)
  # uncomment line below when combine function bug is fixed for Windows
  #registerDoParallel(cores=detectCores())
} else {
  library(doMC)
  registerDoMC(cores=detectCores())
}
```

- In general, parallel processing with `foreach` works correctly in both Windows and Linux
- The function apply.paramset has a bug in the combine function used with `foreach` which generates an error on Windows

lnz portfolio/account, perform walk forward analysis

```r
rm.strat("opt")

initPortf(name="opt", stock.st, initDate=initDate)
initAcct(name="opt", portfolios="opt",
         initDate=initDate, initEq=initEq)
initOrders(portfolio="opt", initDate=initDate)
```

```r
results <- walk.forward(
  strategy.st=strat.st,
  paramset.label='BBOPT',
  portfolio.st="opt",
  account.st="opt",
  period='years',
  k.training=4,
  k.testing=1,
  nsamples=0,
  audit.prefix='wfa',
  anchored=FALSE,
  verbose=TRUE
)
```

Console output during run

```
[1] "=== training BBOPT on 2007-01-03/2010-12-31"

[1] "=== testing param.combo 11 on 2011-01-03/2011-12-30"
    n sd
11 10  2



[1] "=== training BBOPT on 2008-01-02/2011-12-30"

[1] "=== testing param.combo 11 on 2012-01-03/2012-12-31"
    n sd
11 10  2



[1] "=== training BBOPT on 2009-01-02/2012-12-31"

[1] "=== testing param.combo 19 on 2013-01-02/2013-12-31"
    n  sd
19 25 2.5
```

WFA results

- After the call to `walk.forward`, the portfolio object is updated with the concatenated results of all of the out-of-sample tests

```r
PerformanceAnalytics:::textplot(t(tradeStats("opt")))
```

```r
txns <- getTxns("opt",stock.st)
txns$Net.Txn.Realized.PL <- round(txns$Net.Txn.Realized.PL)
PerformanceAnalytics:::textplot(head(txns))
```

```r
PerformanceAnalytics:::textplot(tail(txns))
```

WFA trade stats

	USO
Portfolio	opt
Symbol	USO
Num.Txns	72
Num.Trades	36
Net.Trading.PL	−13382
Avg.Trade.PL	−371.72222
Med.Trade.PL	−200
Largest.Winner	6501
Largest.Loser	−10881
Gross.Profits	48202
Gross.Losses	−61584
Std.Dev.Trade.PL	4066.0715
Percent.Positive	38.888889
Percent.Negative	61.111111
Profit.Factor	0.7827033
Avg.Win.Trade	3443
Med.Win.Trade	3654
Avg.Losing.Trade	−2799.2727
Med.Losing.Trade	−1874.5
Avg.Daily.PL	−371.72222
Med.Daily.PL	−200
Std.Dev.Daily.PL	4066.0715
Ann.Sharpe	−1.4512552
Max.Drawdown	−34706
Profit.To.Max.Draw	−0.38558174
Avg.WinLoss.Ratio	1.2299623
Med.WinLoss.Ratio	1.9493198
Max.Equity	13460
Min.Equity	−21246
End.Equity	−13382

WFA Transactions

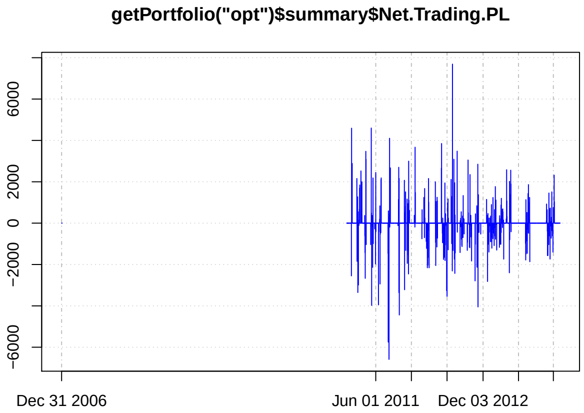
	Txn.Qty	Txn.Price	Txn.Fees	Txn.Value	Txn.Avg.Cost	Net.Txn.Realized.PL
2006−12−31	0	0	0	0	0	0
2011−01−26	2800	36.85	0	103180	36.85	0
2011−01−31	−2800	38.61	0	−108108	38.61	4928
2011−02−23	−2600	39.8	0	−103480	39.8	0
2011−03−11	2600	40.69	0	105794	40.69	−2314
2011−03−16	2500	39.68	0	99200	39.68	0

	Txn.Qty	Txn.Price	Txn.Fees	Txn.Value	Txn.Avg.Cost	Net.Txn.Realized.PL
2013−04−16	3200	31.76	0	101632	31.76	0
2013−04−26	−3200	33.12	0	−105984	33.12	4352
2013−07−08	−2700	36.41	0	−98307	36.41	0
2013−07−31	2700	37.36	0	100872	37.36	−2565
2013−10−23	2800	34.95	0	97860	34.95	0
2013−12−04	−2800	34.88	0	−97664	34.88	−196

- out-of-sample transactions

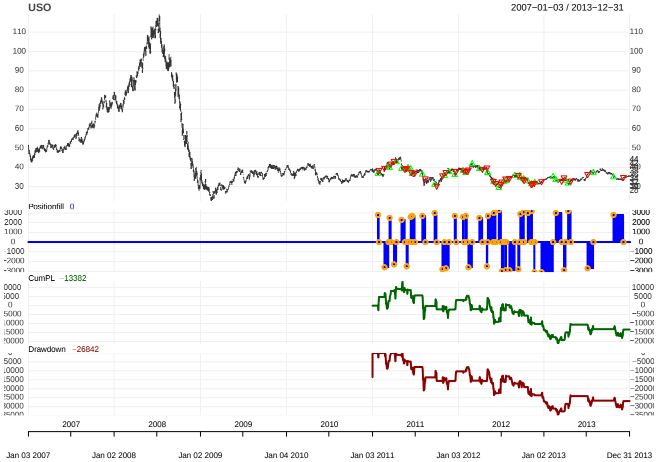
Out-of-sample P&L

```
plot(getPortfolio("opt")$summary$Net.Trading.PL,minor.ticks=FALSE,type="h",col=4)
```



WFA positions

```
chart.Posn("opt",stock.st)
```



Returned results

```
names(results)

## [1] ""             ""             ""             "tradeStats"

names(results[[1]])

## [1] "training.timespan" "apply.paramset"    "testing.timespan"

results[[1]]$training.timespan

## [1] "2007-01-03/2010-12-31"

results[[1]]$testing.timespan

## [1] "2011-01-03/2011-12-30"

names(results[[1]]$apply.paramset)

##  [1] "opt.1"      "tradeStats" "opt.2"      "opt.3"      "opt.4"
##  [6] "opt.5"      "opt.6"      "opt.7"      "opt.8"      "opt.9"
## [11] "opt.10"     "opt.11"     "opt.12"     "opt.13"     "opt.14"
## [16] "opt.15"     "opt.16"     "opt.17"     "opt.18"     "opt.19"
## [21] "opt.20"     "opt.21"     "opt.22"     "opt.23"     "opt.24"
## [26] "opt.25"
```

Trade stats for each training period

```
idx <- which.max(results[[1]]$apply.paramset$tradeStats$Net.Trading.PL)
results[[1]]$apply.paramset$tradeStats[idx,1:7]

##     n sd Portfolio Symbol Num.Txns Num.Trades Net.Trading.PL
## 11 10  2    opt.11    USO      154         77          84148

idx <- which.max(results[[2]]$apply.paramset$tradeStats$Net.Trading.PL)
results[[2]]$apply.paramset$tradeStats[idx,1:7]

##     n sd Portfolio Symbol Num.Txns Num.Trades Net.Trading.PL
## 11 10  2    opt.11    USO      142         71          92924

idx <- which.max(results[[3]]$apply.paramset$tradeStats$Net.Trading.PL)
results[[3]]$apply.paramset$tradeStats[idx,1:7]

##     n  sd Portfolio Symbol Num.Txns Num.Trades Net.Trading.PL
## 19 25 2.5    opt.19    USO       46         23          56803
```

Files generated by walk.forward

- For each training set, a separate file is created, containing an enviroment called .audit, with all in-sample portfolios and orderbooks as well as information as to which param.combos were evaluated, and the result of the objective function.
- In addition, a special file is generated that contains portfolio and orderbook for the concatenated testing param.combos as selected by the objective function, plus (optionally) complete in-sample portfolios and orderbooks for reference.

```
list.files(pattern="^wfa.*\\.RData$")

## [1] "wfa.results.RData"
## [2] "wfa.USO.2007-01-03.2010-12-31.RData"
## [3] "wfa.USO.2008-01-02.2011-12-30.RData"
## [4] "wfa.USO.2009-01-02.2012-12-31.RData"
```

Files generated by walk.forward

- File <audit.prefix>.<symbol>.<start>.<end>.RData contains:
  - .audit - contains all in-sample portfolios and orderbooks
  - .blotter (empty)
  - .strategy (empty)
- File <audit.prefix>.results.RData contains:
  - .audit - contains all in-sample portfolios and orderbooks
  - .blotter (empty)
  - .strategy (empty)

Training file contents

```
> load("~/RProjects/UW/CFRM551/WFA/wfa.USO.2007-01-03.2010-12-31.RData")

> ls(all=TRUE)
[1] ".audit"       ".blotter"     ".Random.seed" ".strategy"

> ls(.audit)
 [1] "constraints"        "distributions"      "obj.func"           "order_book.opt.1"   "order_book.opt.10"
 [6] "order_book.opt.11" "order_book.opt.12" "order_book.opt.13" "order_book.opt.14" "order_book.opt.15"
[11] "order_book.opt.16" "order_book.opt.17" "order_book.opt.18" "order_book.opt.19" "order_book.opt.2"
[16] "order_book.opt.20" "order_book.opt.21" "order_book.opt.22" "order_book.opt.23" "order_book.opt.24"
[21] "order_book.opt.25" "order_book.opt.3"  "order_book.opt.4"  "order_book.opt.5"  "order_book.opt.6"
[26] "order_book.opt.7"  "order_book.opt.8"  "order_book.opt.9"  "param.combo"        "param.combo.idx"
[31] "param.combo.nr"    "param.combos"       "paramset.label"     "portfolio.opt.1"    "portfolio.opt.10"
[36] "portfolio.opt.11"  "portfolio.opt.12"  "portfolio.opt.13"  "portfolio.opt.14"  "portfolio.opt.15"
[41] "portfolio.opt.16"  "portfolio.opt.17"  "portfolio.opt.18"  "portfolio.opt.19"  "portfolio.opt.2"
[46] "portfolio.opt.20"  "portfolio.opt.21"  "portfolio.opt.22"  "portfolio.opt.23"  "portfolio.opt.24"
[51] "portfolio.opt.25"  "portfolio.opt.3"   "portfolio.opt.4"   "portfolio.opt.5"   "portfolio.opt.6"
[56] "portfolio.opt.7"   "portfolio.opt.8"   "portfolio.opt.9"   "tradeStats"         "training.timespan"
[61] "user.func"
>
```

Testing file contents

```
> load("~/RProjects/UW/CFRM551/WFA/wfa.results.RData")

> ls(all=TRUE)
[1] ".audit"       ".blotter"     ".Random.seed" ".strategy"

> ls(.audit)
 [1] "account.opt"        "constraints"        "distributions"      "order_book.opt"     "order_book.opt.1"
 [6] "order_book.opt.10" "order_book.opt.11" "order_book.opt.12" "order_book.opt.13" "order_book.opt.14"
[11] "order_book.opt.15" "order_book.opt.16" "order_book.opt.17" "order_book.opt.18" "order_book.opt.19"
[16] "order_book.opt.2"  "order_book.opt.20" "order_book.opt.21" "order_book.opt.22" "order_book.opt.23"
[21] "order_book.opt.24" "order_book.opt.25" "order_book.opt.3"  "order_book.opt.4"  "order_book.opt.5"
[26] "order_book.opt.6"  "order_book.opt.7"  "order_book.opt.8"  "order_book.opt.9"  "param.combos"
[31] "paramset.label"    "portfolio.opt"     "portfolio.opt.1"   "portfolio.opt.10"  "portfolio.opt.11"
[36] "portfolio.opt.12"  "portfolio.opt.13"  "portfolio.opt.14"  "portfolio.opt.15"  "portfolio.opt.16"
[41] "portfolio.opt.17"  "portfolio.opt.18"  "portfolio.opt.19"  "portfolio.opt.2"   "portfolio.opt.20"
[46] "portfolio.opt.21"  "portfolio.opt.22"  "portfolio.opt.23"  "portfolio.opt.24"  "portfolio.opt.25"
[51] "portfolio.opt.3"   "portfolio.opt.4"   "portfolio.opt.5"   "portfolio.opt.6"   "portfolio.opt.7"
[56] "portfolio.opt.8"   "portfolio.opt.9"   "tradeStats"        "user.func"
>
```

chart.forward.training and chart.forward

The `chart.forward.training` function plots the cummulative net profit and drawdown for each parameter combination portfolio in a training period.

The `chart.forward` function plots the cummulative net profit and drawdown for each parameter combination portfolio along with the optimal portfolio used in the testing period.

Usage:

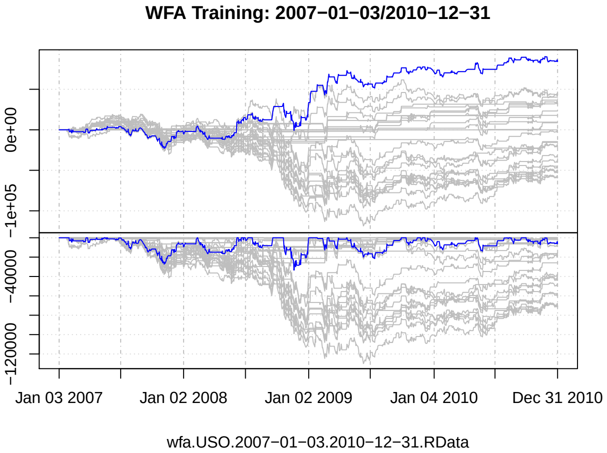
```
chart.forward.training(audit.filename)
chart.forward(audit.filename)
```

Main arguments:

audit.filename	name of .audit environment file as produced by walk.forward()

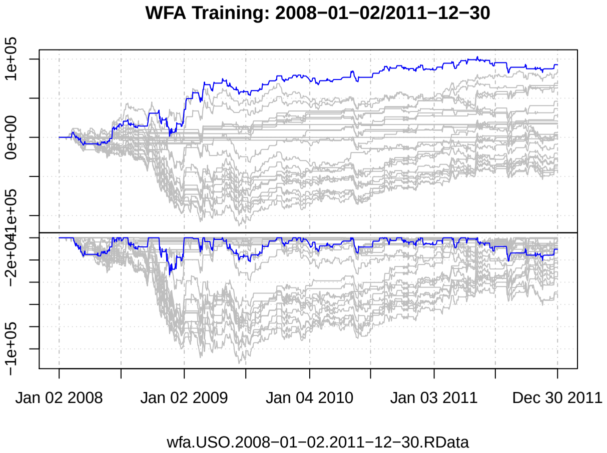
WFA training period performance

```
chart.forward.training("wfa.USO.2007-01-03.2010-12-31.RData")
```



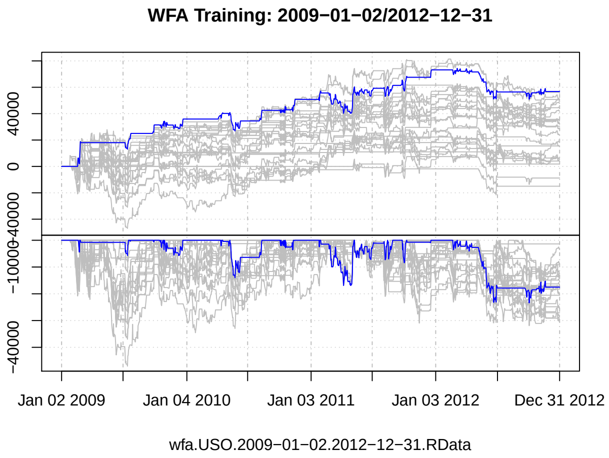
WFA training period performance

```
chart.forward.training("wfa.USO.2008-01-02.2011-12-30.RData")
```



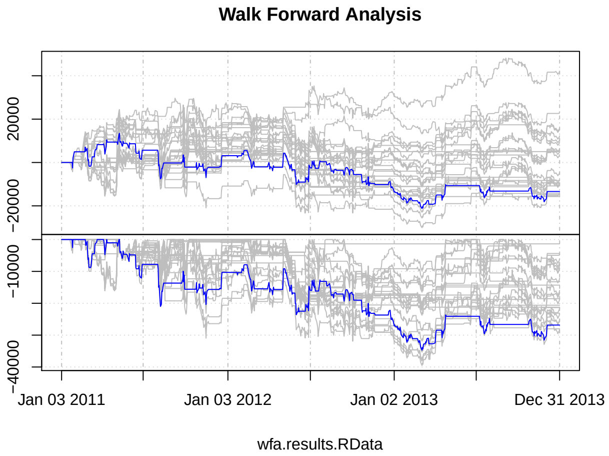
WFA training period performance

```r
chart.forward.training("wfa.USO.2009-01-02.2012-12-31.RData")
```



WFA testing period performance

```
chart.forward("wfa.results.RData")
```



COMPUTATIONAL FINANCE & RISK MANAGEMENT

UNIVERSITY *of* WASHINGTON

Department of Applied Mathematics

`http://depts.washington.edu/compfin`