Pair Trading of GLD and GDX

Example 3.6:

Pair Trading of GLD and GDX

GLD versus GDX is a good candidate for pair trading because GLD reflects the spot price of gold, and GDX is a basket of gold-mining stocks. It makes intuitive sense that their prices should move in tandem.

(also see, e.g., epchan.blogspot.com/2006/11/reader-suggested-possible-trading.html)

We perform a regression analysis on the training set to determine the hedge ratio between GLD and GDX, and then define entry and exit thresholds for a pair-trading strategy.

We will see how optimiz- ing these thresholds on the training set changes the performance on the test set.

```
library(quantmod)
```

```
## Loading required package: xts
```

```
## Warning: package 'xts' was built under R version 3.4.4
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: TTR
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
smbls <- c("GLD", "GDX")
getSymbols(smbls, auto.assign = T)
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).
```

```
## [1] "GLD" "GDX"
```

```
n <- length(GDX$GDX.Adjusted)
upper <- round(length(GDX$GDX.Adjusted)*.8)
index(GDX[upper,])
```

```
## [1] "2016-01-13"
```

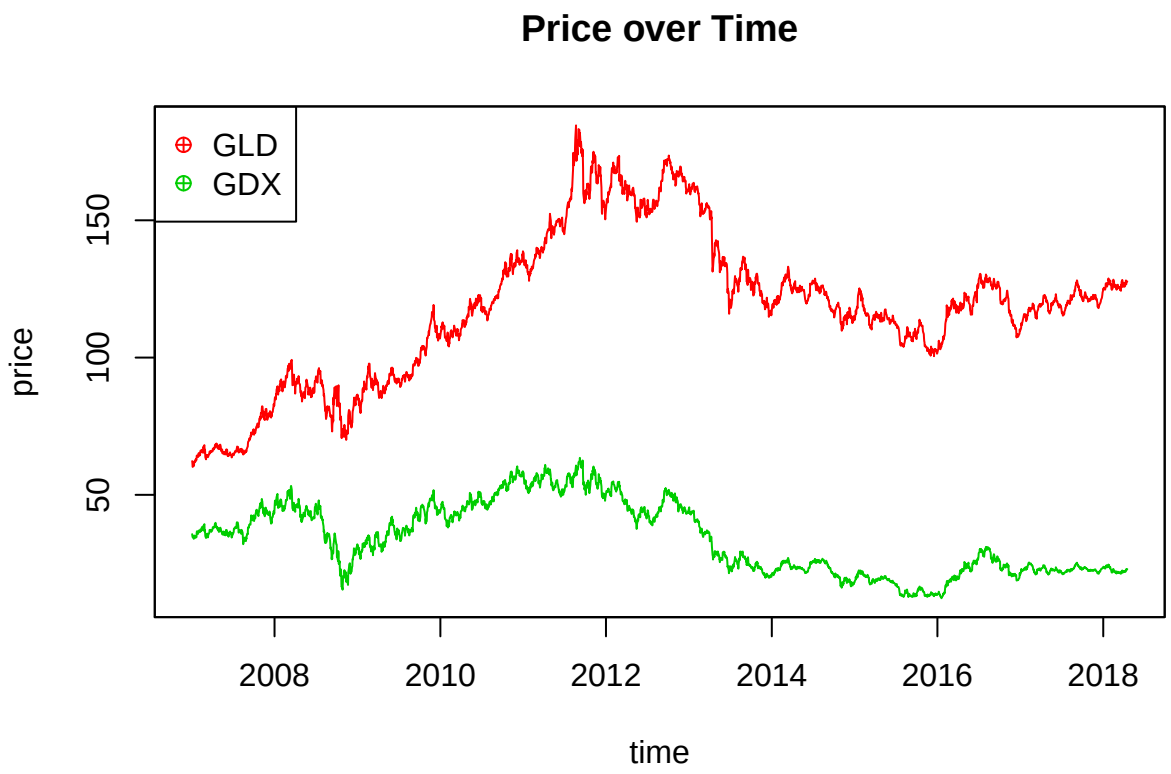
```
train <- 1:upper
test <- (upper+1):n

gold_prices <- merge(GLD$GLD.Adjusted, GDX$GDX.Adjusted)
colnames(gold_prices) <- smbls

gold_lret <- diff(log(gold_prices))
gold_lret_train <- gold_lret[train,]
gold_lret_test <- gold_lret[test,]

head(gold_lret_train)
```

```
##                      GLD          GDX
## 2007-01-03           NA           NA
## 2007-01-04 -0.010167069 -0.018648054
## 2007-01-05 -0.024299406 -0.013613062
## 2007-01-08  0.005138876  0.002415790
## 2007-01-09  0.006099055 -0.006724952
## 2007-01-10 -0.004281924 -0.011946610
```

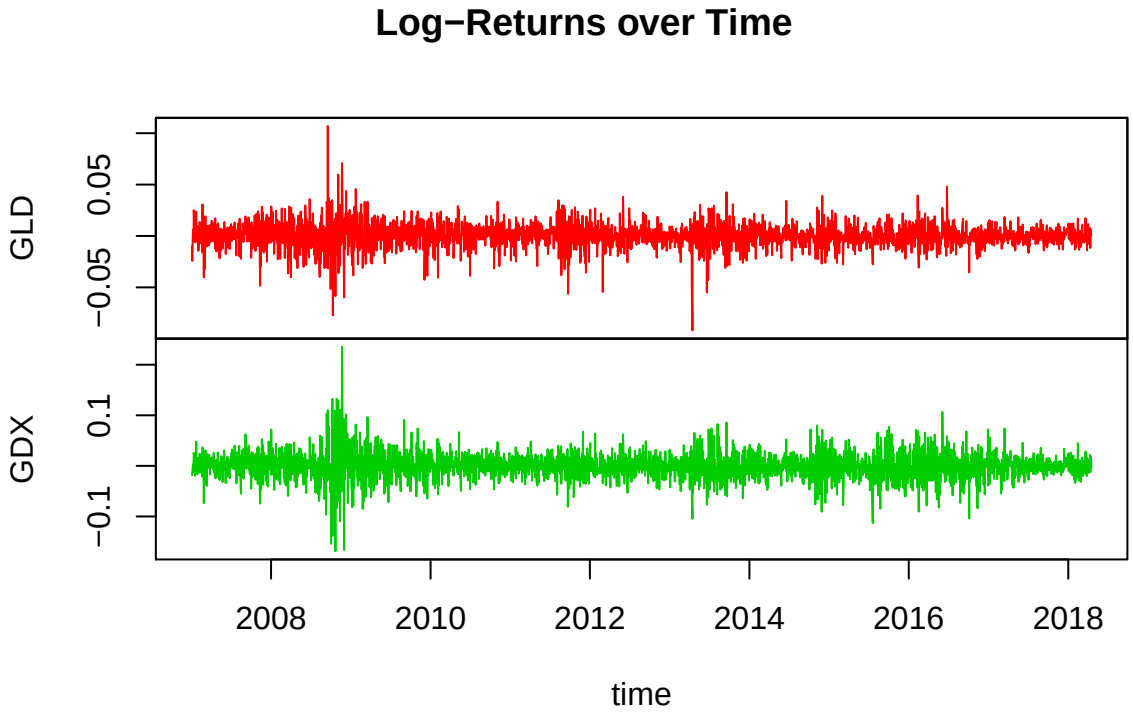
```
head(gold_lret_test)
```

```
##                       GLD          GDX
## 2016-01-14 -0.0163670161 -0.035905077
## 2016-01-15  0.0102367384 -0.003051072
## 2016-01-19 -0.0009612516 -0.048522830
## 2016-01-20  0.0132793923  0.030017972
## 2016-01-21  0.0011381487  0.004658377
## 2016-01-22 -0.0046557926  0.009252232
```

```
plot.zoo(gold_prices, plot.type = "single", col=2:3, ylab = "price", xlab = "time", main = "Price over T
legend("topleft", smbls, pch = 10, col = 2:3)
```

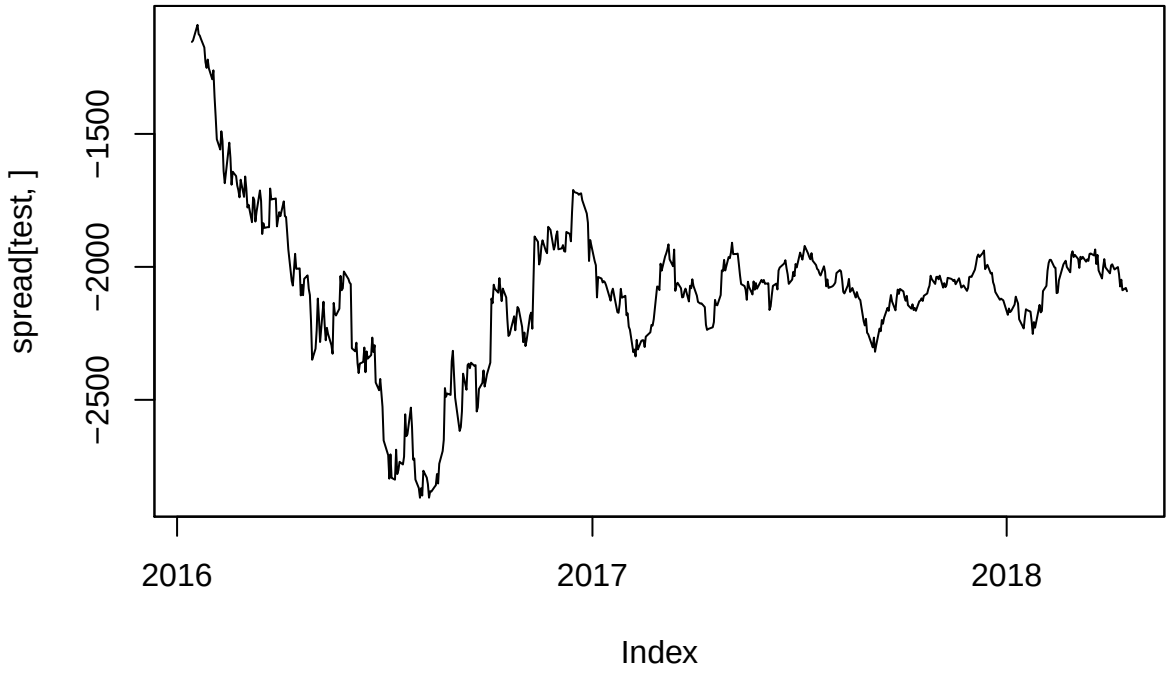



```
plot.zoo(gold_lret, plot.type = "multiple", col=2:3, xlab = "time", main = "Log-Returns over Time")
```




```
results <- lm(gold_prices$GLD ~ gold_prices$GDX)
hedge_ratio <- results$coefficients[1]
spread=gold_prices$GLD-hedge_ratio*gold_prices$GDX

plot.zoo(spread[train,])
```

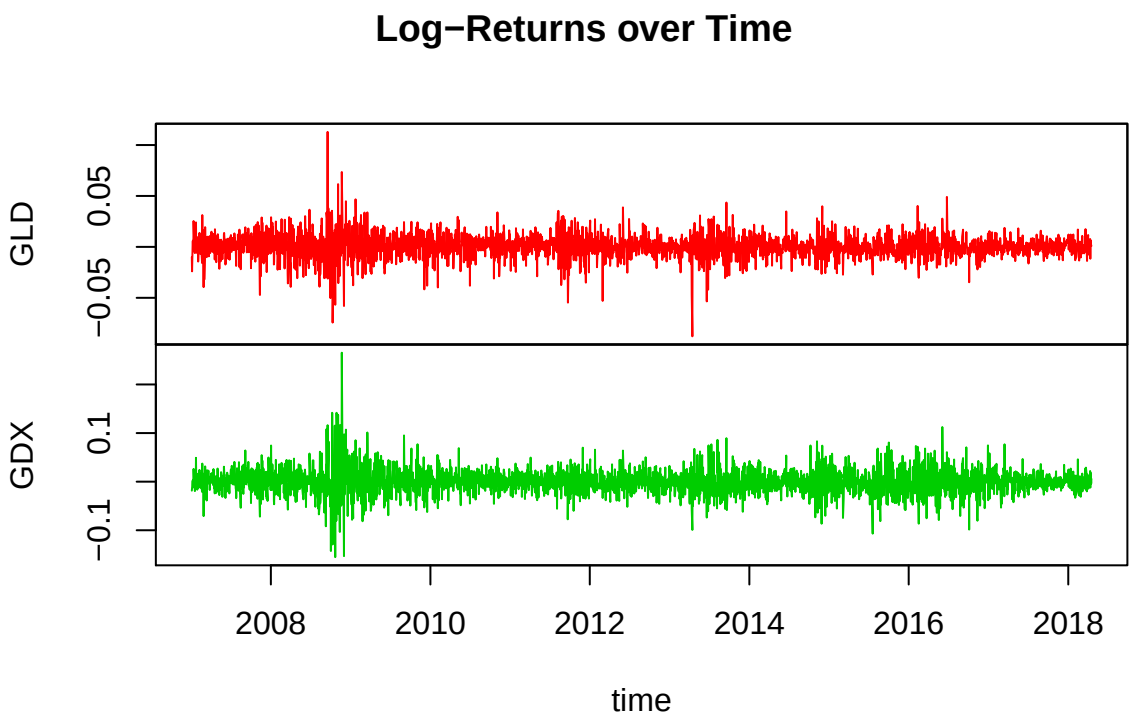

```
plot.zoo(spread[test,])
```




```
spread_mean <- mean(spread[train,]);
spread_sd <- sd(spread[train,]);
zscore <- (spread - spread_mean)/spread_sd;

# buy spread when its value drops below 2 standard deviations.
longs <- zscore <= -1

# short spread when its value rises above 2 standard deviations.
shorts <- zscore >= 1
```

```
# exit any spread position when its value is within 1 standard deviation of its mean.
exits <- abs(zscore) <= 0.5

# initialize positions array
pos <- xts(rep(NA, n), order.by = index(gold_prices))
positions <- merge(pos, pos)
colnames(positions) <- smbls

# long entries
positions[shorts, ] <- matrix(rep(c(-1,1), length(which(shorts))), ncol =2, byrow = T)
# short entries
positions[longs,  ] <- matrix(rep(c(-1,1), length(which(longs))), ncol =2, byrow = T)
# View(positions["2010-11/12"])
# View(longs["2010-11/12"])

# exit positions
positions[exits,  ] <- matrix(rep(c(0, 0), length(which(exits))), ncol =2, byrow = T)

positions = na.locf(positions)
```

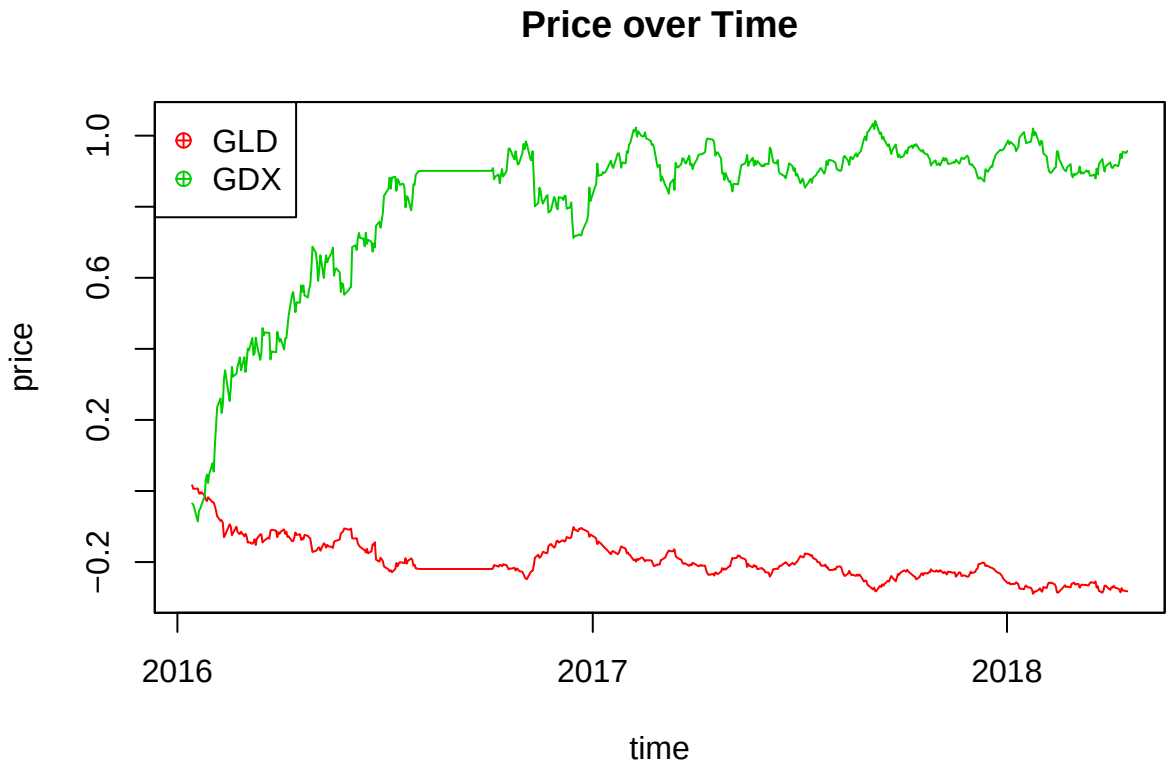
```
daily_ret <- (gold_prices - lag(gold_prices))/lag(gold_prices)
daily_ret <- daily_ret[-1]

plot.zoo(daily_ret, plot.type = "multiple", col=2:3, xlab = "time", main = "Log-Returns over Time")
```




```
pnl <- lag(positions)*daily_ret

# the Sharpe ratio on the training set should be about 2.3
sharpe_train <- sqrt(252)*mean(pnl[train[-1],]/sd(pnl[train[-1],]))
sharpe_train
```

```
## [1] -0.1072002
```

```
# the Sharpe ratio on the test set should be about 1.5
sharpe_testset=sqrt(252)*mean(pnl[test-1,])/sd(pnl[test-1,])
plot.zoo(cumsum(pnl[test-1,]), plot.type = "single", col=2:3, ylab = "price", xlab = "time", main = "Pri
legend("topleft", smbls, pch = 10, col = 2:3)
```




```
sharpe_testset
```

```
## [1] 0.5555997
```