Tony Au (Github: restinghouse0203)

Pair Trading with Time-Series Model: BHP vs VALE

1. Data Preparation

```
# Load necessary libraries and data
library(urca)
bhp = read.table("d-bhp0206.txt", header=T)
vale = read.table("d-vale0206.txt", header=T)
```

We perform a logarithmic transformation on the adjusted closing prices to stabilize variance.

```
# Log transform the adj. close price
bhp = log(bhp[,9])
vale = log(vale[,9])
```

2. Preliminary Analysis

2.1 Least-Squares Estimation

We begin by estimating the static relationship between BHP and VALE using Ordinary Least Squares (OLS).

```
m1 = lm(bhp~vale)
summary(m1)
```

```
##
## Call:
## lm(formula = bhp ~ vale)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.151818 -0.028265  0.003121  0.029803  0.147105
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.822648   0.003662   497.7   <2e-16 ***
## vale        0.716664   0.002354   304.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04421 on 944 degrees of freedom
## Multiple R-squared:  0.9899, Adjusted R-squared:  0.9899
## F-statistic: 9.266e+04 on 1 and 944 DF,  p-value: < 2.2e-16
```

The estimated static relationship is:

$$\widehat{\text{bhp}} = 1.823 + 0.717 \cdot \text{vale} + \varepsilon, \quad \varepsilon \sim N(0, 0.044^2)$$

Interpretation: The coefficient indicates that for every 1% increase in VALE, BHP increases by approximately 0.717% on average. The model explains 98.99% of the variability in BHP ($R^2 = 0.9899$), suggesting a strong comovement.

2.2 Residual Analysis (AR(2))

We analyze the residuals of the OLS model to check for stationarity and serial correlation.

```
library(tseries)
wt = m1$residuals
m3 = arima(wt,order=c(2,0,0),include.mean = F)
m3
```

```
##
## Call:
## arima(x = wt, order = c(2, 0, 0), include.mean = F)
##
## Coefficients:
##          ar1     ar2
##       0.8051  0.1219
## s.e.  0.0322  0.0325
##
## sigma^2 estimated as 0.0003326:  log likelihood = 2444.76,  aic = -4883.51
```

```
# ADF test for wt stationarity
adf.test(wt, k=2)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  wt
## Dickey-Fuller = -6.0306, Lag order = 2, p-value = 0.01
## alternative hypothesis: stationary
```

```
# Characteristic roots decomposition
p1=c(1,-m3$coef)
print("The characteristic roots are")
```

```
## [1] "The characteristic roots are"
```

```
1/Mod(polyroot(p1))
```

```
## [1] 0.9353661 0.1302870
```

The error process is modeled as:

$$w_t = 0.8051\, w_{t-1} + 0.1219\, w_{t-2} + \varepsilon_t = (1 - 0.935B)(1 - 0.13B)w_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, 0.0003326)$$

Interpretation: The modulus of both characteristic roots is less than 1, and the Augmented Dickey-Fuller (ADF) test rejects the null hypothesis of a unit root. This confirms that the residual series w_t is stationary, a prerequisite for cointegration. The current value of w_t depends strongly on its immediate lag (0.805) and weakly on the second lag (0.122).

3. Cointegration Analysis

3.1 Johansen Cointegration Test

We apply the Johansen procedure to test for the number of cointegrating relationships.

```
# install.packages("HDTSA")
library(HDTSA)
xt = cbind(bhp,vale)

# Cointegrating relations
cot=ca.jo(xt,ecdet = "const",type = "trace",K=2,spec = "transitory")
summary(cot)
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: trace statistic , without linear trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1]  4.148282e-02  8.206470e-03 -4.210618e-18
##
## Values of teststatistic and critical values of test:
##
##          test 10pct  5pct  1pct
## r <= 1 |  7.78  7.52  9.24 12.97
## r = 0  | 47.77 17.85 19.96 24.60
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##              bhp.l1     vale.l1  constant
## bhp.l1     1.000000  1.0000000  1.000000
## vale.l1   -0.717704 -0.7327542  2.047274
## constant -1.828460 -1.5411890 -5.712629
##
## Weights W:
## (This is the loading matrix)
##
```

```
##             bhp.l1      vale.l1       constant
## bhp.d  -0.06731196 0.004568985 -7.703496e-18
## vale.d  0.02545606 0.007541565  3.627982e-18
```

```
co1=ca.jo(xt,ecdet = "const",type = "eigen",K=2,spec = "transitory")
summary(co1)
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1]  4.148282e-02  8.206470e-03 -4.210618e-18
##
## Values of teststatistic and critical values of test:
##
##          test 10pct  5pct  1pct
## r <= 1 |  7.78  7.52  9.24 12.97
## r = 0  | 40.00 13.75 15.67 20.20
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##             bhp.l1     vale.l1  constant
## bhp.l1     1.000000   1.0000000  1.000000
## vale.l1   -0.717704  -0.7327542  2.047274
## constant -1.828460 -1.5411890 -5.712629
##
## Weights W:
## (This is the loading matrix)
##
##             bhp.l1      vale.l1       constant
## bhp.d  -0.06731196 0.004568985 -7.703496e-18
## vale.d  0.02545606 0.007541565  3.627982e-18
```

Interpretation: Both the trace and maximal eigenvalue tests indicate a cointegration rank of $r = 1$. This confirms a single long-run equilibrium relationship:

$$\text{bhp} \approx 0.718 \times \text{vale} + 1.83$$

This relationship is consistent with the initial OLS estimate.

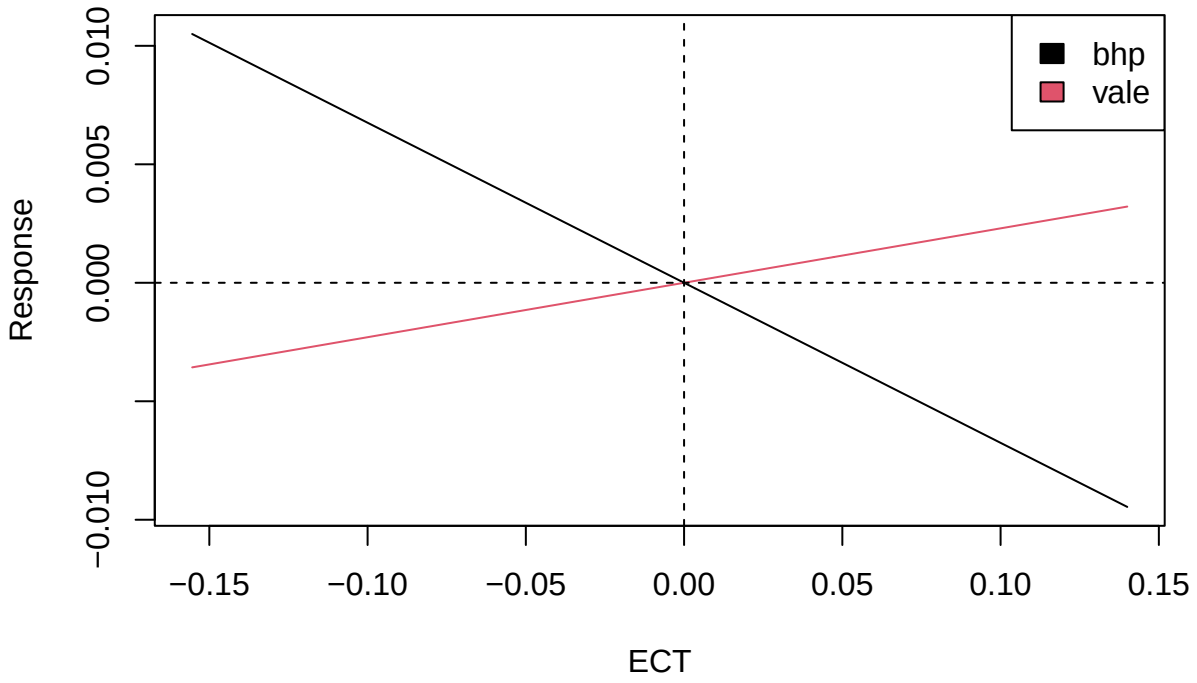
3.2 Vector Error Correction Model (VECM)

We estimate a VECM to understand the adjustment dynamics.

```r
# install.packages("tsDyn")
library(tsDyn)
library(vars)

# Data setup
xt <- ts(cbind(bhp = bhp, vale = vale))

# Estimate VECM directly
vecm <- VECM(xt,
             lag       = 2,              # lags in differences
             r         = 1,              # cointegrating rank
             include   = "const",        # constant inside cointegration relation
             estim     = "ML",           # estimation method
             LRinclude = "const")        # constant location

# Summary
summary(vecm)
```

```
## #############
## ###Model VECM
## #############
## Full sample size: 946    End sample size: 943
## Number of variables: 2    Number of estimated slope parameters 10
## AIC -14851.66    BIC -14798.32    SSR 0.8210436
## Cointegrating vector (estimated by ML):
##    bhp       vale      const
## r1   1 -0.7172022 -1.829477
##
##
##                ECT                bhp -1              vale -1
## Equation bhp  -0.0675(0.0148)*** -0.1102(0.0374)**  0.0724(0.0325)*
## Equation vale 0.0230(0.0171)      0.0662(0.0433)     0.0493(0.0376)
##                bhp -2              vale -2
## Equation bhp  -0.0062(0.0371)     -0.0153(0.0322)
## Equation vale 0.0134(0.0430)      -0.0688(0.0373).
```

```r
# Plot of the cointegration relation
plot_ECT(vecm,
         add.legend     = TRUE,
         legend.location = "topright")
```



Model Structure

Matrix form:

$$\begin{pmatrix}\Delta \text{bhp}_t \\ \Delta \text{vale}_t\end{pmatrix} = \alpha(\beta' z_{t-1}) + \Gamma_1 \begin{pmatrix}\Delta \text{bhp}_{t-1} \\ \Delta \text{vale}_{t-1}\end{pmatrix} + \Gamma_2 \begin{pmatrix}\Delta \text{bhp}_{t-2} \\ \Delta \text{vale}_{t-2}\end{pmatrix} + \varepsilon_t$$

Matrix form (with parameters):

Let $w_t = \text{BHP}_t - 0.717 \cdot \text{VALE}_t$ and $\mu = E(w_t) = 1.829$.

$$\begin{pmatrix}\Delta \text{bhp}_t \\ \Delta \text{vale}_t\end{pmatrix} = \begin{pmatrix}-0.0675 \\ 0.0230\end{pmatrix} (w_{t-1} - 1.829) + \begin{pmatrix}-0.1102 & 0.0724 \\ 0.0662 & 0.0493\end{pmatrix} \begin{pmatrix}\Delta \text{bhp}_{t-1} \\ \Delta \text{vale}_{t-1}\end{pmatrix} + \begin{pmatrix}-0.0062 & -0.0153 \\ 0.0134 & -0.0688\end{pmatrix} \begin{pmatrix}\Delta \text{bhp}_{t-2} \\ \Delta \text{vale}_{t-2}\end{pmatrix} + \varepsilon_t$$

Interpretation: The error correction term for BHP (-0.0675) is statistically significant, indicating that BHP adjusts towards the equilibrium at a speed of approximately 6.75% per period. In contrast, the adjustment coefficient for VALE is small and insignificant, suggesting that VALE is weakly exogenous and acts as the driver in this pair.

4. Statistical Arbitrage Strategy

4.1 Methodology

We implement a mean-reversion strategy based on the spread $w_t = \text{BHP}_t - 0.717 \cdot \text{VALE}_t$, with mean $\mu = 1.829$ and standard deviation σ_w.

Trading Rules:

- **Upper Threshold:** $\mu + \sigma_w$
- **Lower Threshold:** $\mu - \sigma_w$
- **Signal:**
  - If $w_t > \mu + \sigma_w$: Short 1 unit of BHP, Long 0.717 units of VALE (betting spread decreases).
  - If $w_t < \mu - \sigma_w$: Long 1 unit of BHP, Short 0.717 units of VALE (betting spread increases).
- **Exit:** When w_t reverts to the mean μ.

4.2 Implementation

We compute the spread and the trading thresholds.

```
# Compute spread w_t
gamma <- 0.717 # w_t := bhp_t - \gamma*vale_t
mu <- 1.829 # E(w_t)
w_t <- xt[, "bhp"] - gamma * xt[, "vale"]

# Standard deviation of w_t
sd_w <- sd(w_t)

# Residuals from VECM
res <- residuals(vecm)
sd_eps_bhp <- sd(res[, 1])
sd_eps_vale <- sd(res[, 2])

# Output statistics
cat("SD of w_t:", sd_w, "\n")
```

```
## SD of w_t: 0.04418214
```

```
cat("SD of  _bhp:", sd_eps_bhp, "\n")
```

```
## SD of  _bhp: 0.01926604
```

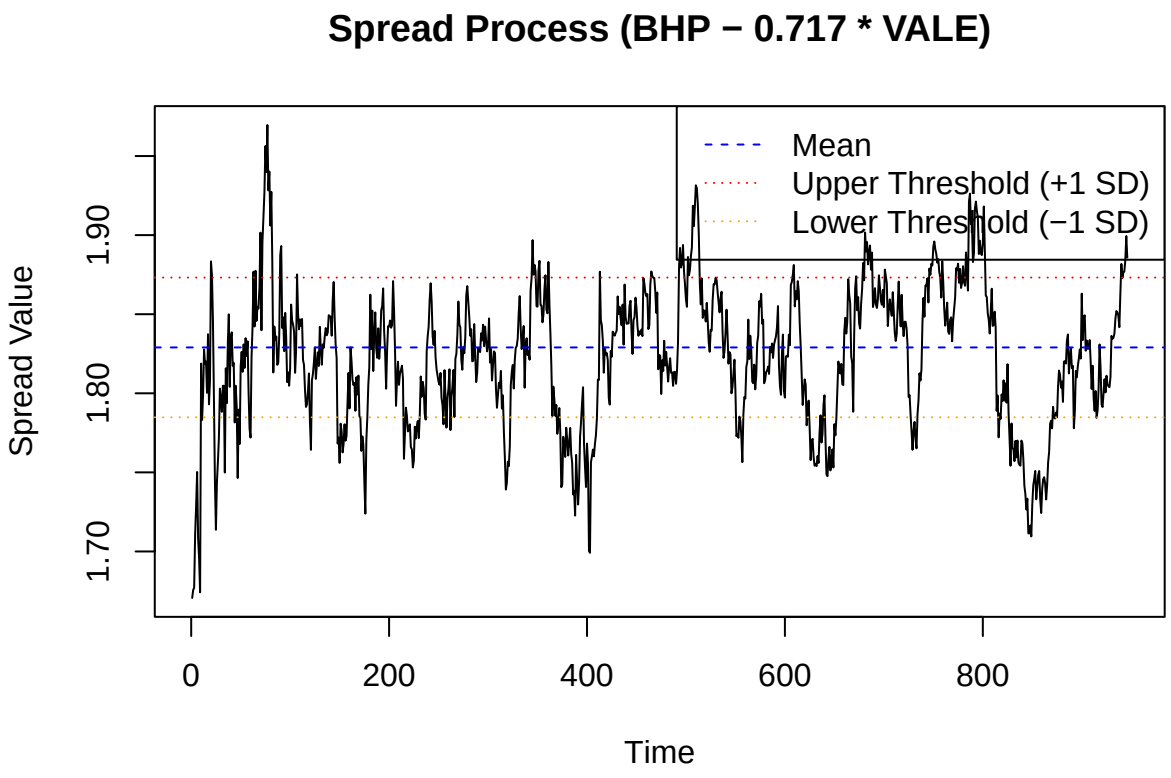
```
cat("SD of  _vale:", sd_eps_vale, "\n")
```

```
## SD of  _vale: 0.02224961
```

The spread process and thresholds are visualized below:

```
# Plot parameters
gamma <- 0.717
mu <- 1.829
w_t <- xt[, "bhp"] - gamma * xt[, "vale"]
sd_w <- sd(w_t)
upper <- mu + sd_w
lower <- mu - sd_w

# Plot spread
```

```
plot(w_t, type="l", main="Spread Process (BHP - 0.717 * VALE)", ylab="Spread Value", xlab="Time")
abline(h=mu, col="blue", lty=2)
abline(h=upper, col="red", lty=3)
abline(h=lower, col="orange", lty=3)
legend("topright", c("Mean", "Upper Threshold (+1 SD)", "Lower Threshold (-1 SD)"),
       col=c("blue","red","orange"), lty=c(2,3,3))
```



4.3 Backtest Results

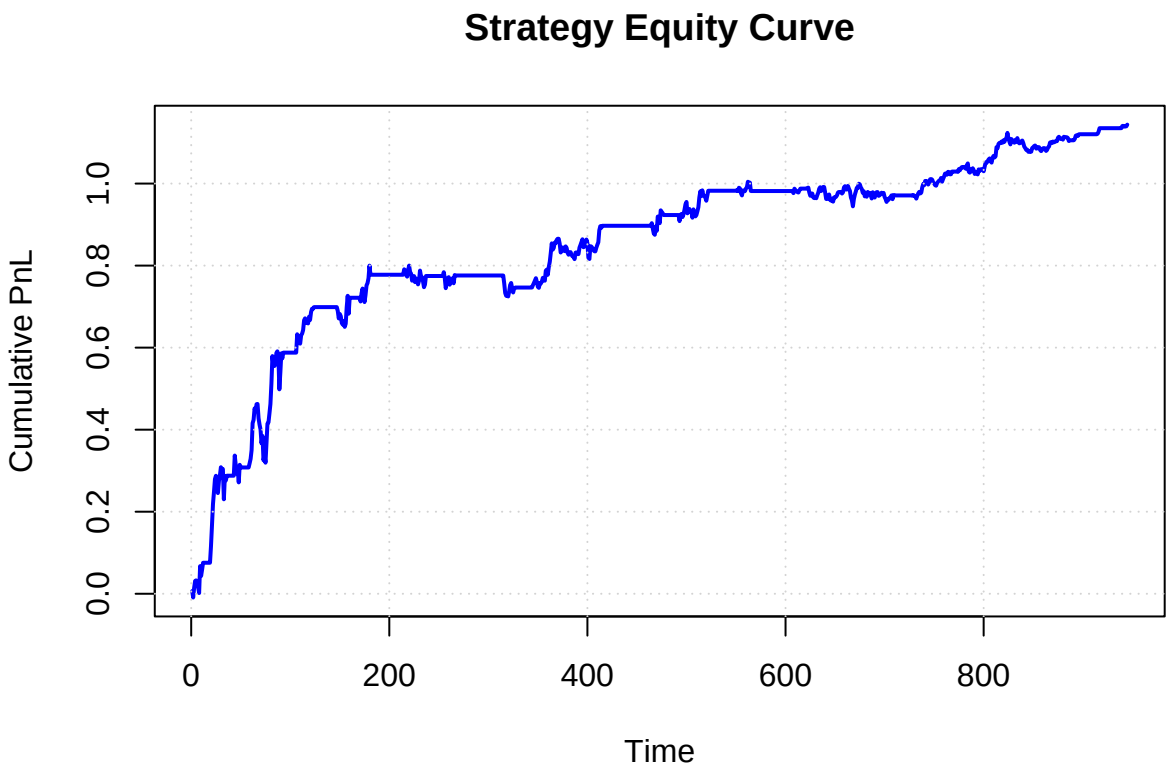
We backtest the strategy over the sample period.

```
# Backtest logic
positions <- rep(0, length(w_t))  # 1: long spread, -1: short spread, 0: neutral
for (t in 2:length(w_t)) {
  if (w_t[t-1] > upper) positions[t] <- -1
  else if (w_t[t-1] < lower) positions[t] <- 1
  else if (positions[t-1] != 0 && abs(w_t[t-1] - mu) < 0.1 * sd_w) positions[t] <- 0  # exit near mean
  else positions[t] <- positions[t-1]
}

# Returns (assume daily log returns)
ret_bhp <- diff(log(xt[, "bhp"]))
ret_vale <- diff(log(xt[, "vale"]))
strat_ret <- positions[2:length(positions)] * (ret_bhp - gamma * ret_vale)
```

```r
# Performance Metrics
cum_pnl <- cumsum(strat_ret)
pnl <- sum(strat_ret)
max_dd <- min(cum_pnl - cummax(cum_pnl))
sharpe <- mean(strat_ret) / sd(strat_ret) * sqrt(252)  # annualized
hit_rate <- sum(strat_ret > 0) / length(strat_ret)

cat("PnL:", pnl, "\nMaxDD:", max_dd, "\nSharpe:", sharpe, "\nHit Rate:", hit_rate)
```

```
## PnL: 1.143625
## MaxDD: -0.1431614
## Sharpe: 1.687619
## Hit Rate: 0.3185185
```

```r
# Equity Curve Visualization
plot(cum_pnl, type = "l", col = "blue", lwd = 2,
     main = "Strategy Equity Curve",
     xlab = "Time", ylab = "Cumulative PnL")
grid()
```



Performance Discussion:

The pairs trading strategy yields a total Profit and Loss (PnL) of 1.14 with an annualized Sharpe ratio of 1.69, indicating a favorable risk-adjusted return profile. However, the hit rate is approxi-

mately 32%, which suggests that the strategy's profitability relies on capturing a smaller number of large mean-reversion moves rather than a high frequency of small wins. The maximum drawdown of approximately 14% highlights the tail risk inherent in spread divergence, emphasizing the need for robust risk management or stop-loss mechanisms in live implementation.