S5261_FINAL

Nguyen Thuy Linh, tn2382

6/9/2019

We choose rolling window size m to be 252. We choose forecast horizon, h, to be 5. Our first rolling window will consist of observations for period 1 from 1 through 252, and the second rolling window will contain observations for period 2 from $1 + 5 = 6$ through $252 + 5 = 257$. We will fit AR(1) and VAR(1) with LASSO models, followed by Black-Litterman which takes the mean and covariances from the mentioned models (priors), and combines with sample mean and covariance (iid likelihood) based on chosen values of tau, 0.8, 1 and 1.2. We will choose the stocks for all models according to the highest beta from regressing stock prices on time. As for equal weights, equal weights 1 means equal weights for the selected 30 assets, and equal weights 2 means equal weights for the whole portfolio (3174) assets. For optimization, we will implement the minimum variance portfolio. We will annualize all results.

Below is the code to import data if the working directory is a folder with all the stocks AND the Rmarkdown file, which is excluded as seen below.

```
# Import data set
file.names <- list.files()
file.names <- file.names[-6398] # exclude the Rmarkdown file name

my_read <- function(x) {
  files <- fread(x, header = TRUE, nrows = 2521, na.strings = "NA")
  symbols <- gsub(".csv", "", x)
  n <- nrow(files)
  if (n == 2521) {
    if (files[1,date] == "2019-04-18" & files[n,date] == "2009-04-15") {
      if ((sum(files[, volume]>1000)/n) >= 0.95 && (sum(files[, adjclose]>0.01)/n >= 0.95)) {
        files <- files[,c(1,7)]
        if (sum(is.na(files[,adjclose]))/n <= 0.05) {
          colnames(files) <- c("date", symbols)
          return(files)
        }
      }
    }
  }
}
stocks.full <- bind_cols(lapply(file.names, my_read))
drop.cols <- grep("^date[0-9]+$", colnames(stocks.full))
stocks.full[, (drop.cols) := NULL]
dim(stocks.full)
```

We have 3174 assets in total. We flip the data to have the oldest data on top. We clean the returns to get rid of outliers.

```
# Reverse the order of dates from oldest to most recent
stocks.full <- stocks.full[dim(stocks.full)[1]:1, ]
# Compute daily log returns
returns <- as.data.frame(sapply(as.data.frame(stocks.full[,2:3175]), function(x) diff(log(x))))
# Detect and replace outliers with estimates more consistent with majority of data
returns.cleaned <- sapply(X = returns, FUN = tsclean)
```

(1)

AR(1) Model

We fit the AR(1) model combined with Black-Litterman (tau = 0.8, 1, 1.2) to obtain optimized weights.

```r
# Now perform the same analysis for all rolling windows

SP500 <- fread("/Users/Linh/Desktop/SP500new.csv", header = T, fill = TRUE)
SP500 <- SP500[,c("Date", "Close")]
SP500 <- SP500[35400:37920,]
SP500.ret <- sapply(as.data.frame(SP500$Close), function(x) diff(log(x)))

# Initialize vectors
rw.count <- 1
returns.opt1 <- vector()
returns.opt2 <- vector()
returns.opt3 <- vector()
returns.eq1 <- vector()
returns.eq2 <- vector()
returns.SP500 <- vector()
names.mat <- matrix(0, nrow = 30, ncol = 453)
res_mat <- matrix(0, nrow = 252, ncol = 30)
forec <- matrix(0, nrow = 5, ncol = 30)
forecasts <- NULL

tau <- c(0.8, 1, 1.2)

weights.all1 <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all1) <- c(names(returns))
weights.all2 <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all2) <- c(names(returns))
weights.all3 <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all3) <- c(names(returns))

w1 <- rep(1/30, 30)
w2 <- rep(1/3174, 3174)

for (i in 1:453) {
  sw <- as.matrix(stocks.full[(1+rw.count):(252+rw.count), 2:3175])
  rw <- returns.cleaned[rw.count:(251+rw.count),]
  # Asset Selection
  mu <- 252*colMeans(sw)
  model <- lm(sw ~ matrix(1:252, nrow = 252, ncol = 1))
  beta <- coef(model)[2,]
  summary441 <- data.frame(cbind(beta,mu))
  rownames(summary441) <- c(names(returns))
  colnames(summary441) <- c("Beta","Mean")
  top30.1.names <- rownames(summary441[order(-summary441$Beta),])[1:30]
  names.mat[,i] <- top30.1.names
  top30.1 <- rw[,c(top30.1.names)]
  mu1 <- matrix(252*colMeans(top30.1), nrow = 30, ncol = 1)
  sd1 <- 252*cov(top30.1)
  for (l in 1:30) {
```

```
  # AR(1)
  fit <- arima(top30.1[,l], order = c(1,0,0))
  forec[,l] <- predict(fit, 5)$pred
  res_mat[,l] <- fit$residuals
}

mu2 <- matrix((252/5)*colSums(forec), nrow = 30, ncol = 1)
sd2 <- 252*cov(res_mat)
forecasts <- rbind(forecasts, forec)

# Black-Litterman mean and covariance
mean_BL1 <- solve(solve(tau[1]*sd1) + solve(sd2)) %*% (solve(tau[1]*sd1) %*% mu1 + solve(sd2) %*% mu2
mean_BL2 <- solve(solve(tau[2]*sd1) + solve(sd2)) %*% (solve(tau[2]*sd1) %*% mu1 + solve(sd2) %*% mu2
mean_BL3 <- solve(solve(tau[3]*sd1) + solve(sd2)) %*% (solve(tau[3]*sd1) %*% mu1 + solve(sd2) %*% mu2
cov_BL1 <- solve(solve(tau[1]*sd1) + solve(sd2))
cov_BL2 <- solve(solve(tau[2]*sd1) + solve(sd2))
cov_BL3 <- solve(solve(tau[3]*sd1) + solve(sd2))

# Portfolio Optimization
Amat1 <- cbind(rep(1, 30), mean_BL1, diag(1, nrow = 30))
Amat2 <- cbind(rep(1, 30), mean_BL2, diag(1, nrow = 30))
Amat3 <- cbind(rep(1, 30), mean_BL3, diag(1, nrow = 30))
muP1 <- seq(min(mean_BL1) + 0.0001, max(mean_BL1) - 0.0001, length = 300)
muP2 <- seq(min(mean_BL2) + 0.0001, max(mean_BL2) - 0.0001, length = 300)
muP3 <- seq(min(mean_BL3) + 0.0001, max(mean_BL3) - 0.0001, length = 300)
sdP1 <- muP1
sdP2 <- muP2
sdP3 <- muP3
weights1 <- matrix(0, nrow = 300, ncol = 30)
weights2 <- matrix(0, nrow = 300, ncol = 30)
weights3 <- matrix(0, nrow = 300, ncol = 30)
for (j in 1:length(muP1)) {
  bvec1 <- c(1, muP1[j], rep(0,30))
  bvec2 <- c(1, muP2[j], rep(0,30))
  bvec3 <- c(1, muP3[j], rep(0,30))
  result1 <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL1)$mat), dvec = rep(0, 30), Amat = Amat1, bvec =
  result2 <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL2)$mat), dvec = rep(0, 30), Amat = Amat2, bvec =
  result3 <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL3)$mat), dvec = rep(0, 30), Amat = Amat3, bvec =
  sdP1[j] <- sqrt(result1$value)
  sdP2[j] <- sqrt(result2$value)
  sdP3[j] <- sqrt(result3$value)
  weights1[j,] <- result1$solution
  weights2[j,] <- result2$solution
  weights3[j,] <- result3$solution
}

# Find minimum variance portfolio
ind1 = (sdP1 == min(sdP1))
ind2 = (sdP2 == min(sdP2))
ind3 = (sdP3 == min(sdP3))
# Print weights of the minimum variance portfolio
final.weights1 <- matrix(weights1[ind1,], ncol = 1)
fw1 <- t(final.weights1)
```

```
  colnames(fw1) <- top30.1.names
  weights.all1 <- smartbind(weights.all1, fw1)
  final.weights2 <- matrix(weights2[ind2,], ncol = 1)
  fw2 <- t(final.weights2)
  colnames(fw2) <- top30.1.names
  weights.all2 <- smartbind(weights.all2, fw2)
  final.weights3 <- matrix(weights3[ind3,], ncol = 1)
  fw3 <- t(final.weights3)
  colnames(fw3) <- top30.1.names
  weights.all3 <- smartbind(weights.all3, fw3)

  returns.opt1[i] <- sum(returns.cleaned[(252+rw.count):(256+rw.count),c(top30.1.names)] %*% final.weig
  returns.opt2[i] <- sum(returns.cleaned[(252+rw.count):(256+rw.count),c(top30.1.names)] %*% final.weig
  returns.opt3[i] <- sum(returns.cleaned[(252+rw.count):(256+rw.count),c(top30.1.names)] %*% final.weig
  # Equal weights 1
  returns.eq1[i] <- sum(returns.cleaned[(252+rw.count):(256+rw.count),c(top30.1.names)] %*% matrix(w1, n
  # Equal weights 2
  returns.eq2[i] <- sum(returns.cleaned[(252+rw.count):(256+rw.count),] %*% matrix(w2, ncol = 1))
  returns.SP500[i] <- sum(SP500.ret[(252+rw.count):(256+rw.count),])

  rw.count <- rw.count + 5
}
```

VAR(1) with LASSO model

We fit the AR(1) model combined with Black-Litterman (tau = 0.8, 1, 1.2) to obtain optimized weights.

```
# Initialize vectors
rw.count.var <- 1
returns1.var <- vector()
returns2.var <- vector()
returns3.var <- vector()
returns.eqvar <- vector()

forecast.var <- matrix(0, nrow = 5, ncol = 30)
forecasts.var <- NULL

weights.all1.var <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all1.var) <- c(names(returns))
weights.all2.var <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all2.var) <- c(names(returns))
weights.all3.var <- matrix(0, nrow = 1, ncol = 3174)
colnames(weights.all3.var) <- c(names(returns))
w.eq.var <- rep(1/30, 30)


for (i in 1:453) {
  rw.var <- returns.cleaned[rw.count.var:(251+rw.count.var),]
  # Asset Selection
  top30.var <- rw.var[,c(names.mat[,i])]
  mu1.var <- matrix(252*colMeans(top30.var), nrow = 30, ncol = 1)
  sd1.var <- 252*cov(top30.var)
```

```r
# Fit VAR
fit.var <- fitVAR(top30.var)
forecast.var <- t(computeForecasts(fit.var, 5))
mu2.var <- (252/5)*colSums(forecast.var)
forecasts.var <- rbind(forecasts.var, forecast.var)
sd2.var <- 252*fit.var$sigma

# Black-Litterman mean and covariance
mean_BL1.var <- solve(solve(tau[1]*sd1.var) + solve(sd2.var)) %*% (solve(tau[1]*sd1.var) %*% mu1.var
mean_BL2.var <- solve(solve(tau[2]*sd1.var) + solve(sd2.var)) %*% (solve(tau[2]*sd1.var) %*% mu1.var
mean_BL3.var <- solve(solve(tau[3]*sd1.var) + solve(sd2.var)) %*% (solve(tau[3]*sd1.var) %*% mu1.var
cov_BL1.var <- solve(solve(tau[1]*sd1.var) + solve(sd2.var))
cov_BL2.var <- solve(solve(tau[2]*sd1.var) + solve(sd2.var))
cov_BL3.var <- solve(solve(tau[3]*sd1.var) + solve(sd2.var))

# Portfolio Optimization
Amat1.var <- cbind(rep(1, 30), mean_BL1.var, diag(1, nrow = 30))
Amat2.var <- cbind(rep(1, 30), mean_BL2.var, diag(1, nrow = 30))
Amat3.var <- cbind(rep(1, 30), mean_BL3.var, diag(1, nrow = 30))
muP1.var <- seq(min(mean_BL1.var) + 0.0001, max(mean_BL1.var) - 0.0001, length = 300)
muP2.var <- seq(min(mean_BL2.var) + 0.0001, max(mean_BL2.var) - 0.0001, length = 300)
muP3.var <- seq(min(mean_BL3.var) + 0.0001, max(mean_BL3.var) - 0.0001, length = 300)
sdP1.var <- muP1.var
sdP2.var <- muP2.var
sdP3.var <- muP3.var
weights1.var <- matrix(0, nrow = 300, ncol = 30)
weights2.var <- matrix(0, nrow = 300, ncol = 30)
weights3.var <- matrix(0, nrow = 300, ncol = 30)
for (j in 1:length(muP1.var)) {
  bvec1.var <- c(1, muP1.var[j], rep(0,30))
  bvec2.var <- c(1, muP2.var[j], rep(0,30))
  bvec3.var <- c(1, muP3.var[j], rep(0,30))
  result1.var <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL1.var)$mat), dvec = rep(0, 30), Amat = Amat
  result2.var <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL2.var)$mat), dvec = rep(0, 30), Amat = Amat
  result3.var <- solve.QP(Dmat = 2*as.matrix(nearPD(cov_BL3.var)$mat), dvec = rep(0, 30), Amat = Amat
  sdP1.var[j] <- sqrt(result1.var$value)
  sdP2.var[j] <- sqrt(result2.var$value)
  sdP3.var[j] <- sqrt(result3.var$value)
  weights1.var[j,] <- result1.var$solution
  weights2.var[j,] <- result2.var$solution
  weights3.var[j,] <- result3.var$solution
}

# Find minimum variance portfolio
ind1.var = (sdP1.var == min(sdP1.var))
ind2.var = (sdP2.var == min(sdP2.var))
ind3.var = (sdP3.var == min(sdP3.var))

# Print weights of the minimum variance portfolio
final.weights1.var <- matrix(weights1.var[ind1.var,], ncol = 1)
fw1.var <- t(final.weights1.var)
colnames(fw1.var) <- names.mat[,i]
weights.all1.var <- smartbind(weights.all1.var, fw1.var)
```

```
  final.weights2.var <- matrix(weights2.var[ind2.var,], ncol = 1)
  fw2.var <- t(final.weights2.var)
  colnames(fw2.var) <- names.mat[,i]
  weights.all2.var <- smartbind(weights.all2.var, fw2.var)

  final.weights3.var <- matrix(weights3.var[ind3.var,], ncol = 1)
  fw3.var <- t(final.weights3.var)
  colnames(fw3.var) <- names.mat[,i]
  weights.all3.var <- smartbind(weights.all3.var, fw3.var)

  returns1.var[i] <- sum(returns.cleaned[(252+rw.count.var):(256+rw.count.var),c(names.mat[,i])] %*% fi
  returns2.var[i] <- sum(returns.cleaned[(252+rw.count.var):(256+rw.count.var),c(names.mat[,i])] %*% fi
  returns3.var[i] <- sum(returns.cleaned[(252+rw.count.var):(256+rw.count.var),c(names.mat[,i])] %*% fi
  # Equal weights 1
  returns.eqvar[i] <- sum(returns.cleaned[(252+rw.count.var):(256+rw.count.var),c(names.mat[,i])] %*% ma
  rw.count.var <- rw.count.var + 5
}
```

We create a vector with dates for the rolling window.

```
dates <- stocks.full[253:2521,1]
dates.rw <- dates[seq(5, nrow(dates), 5)]
```

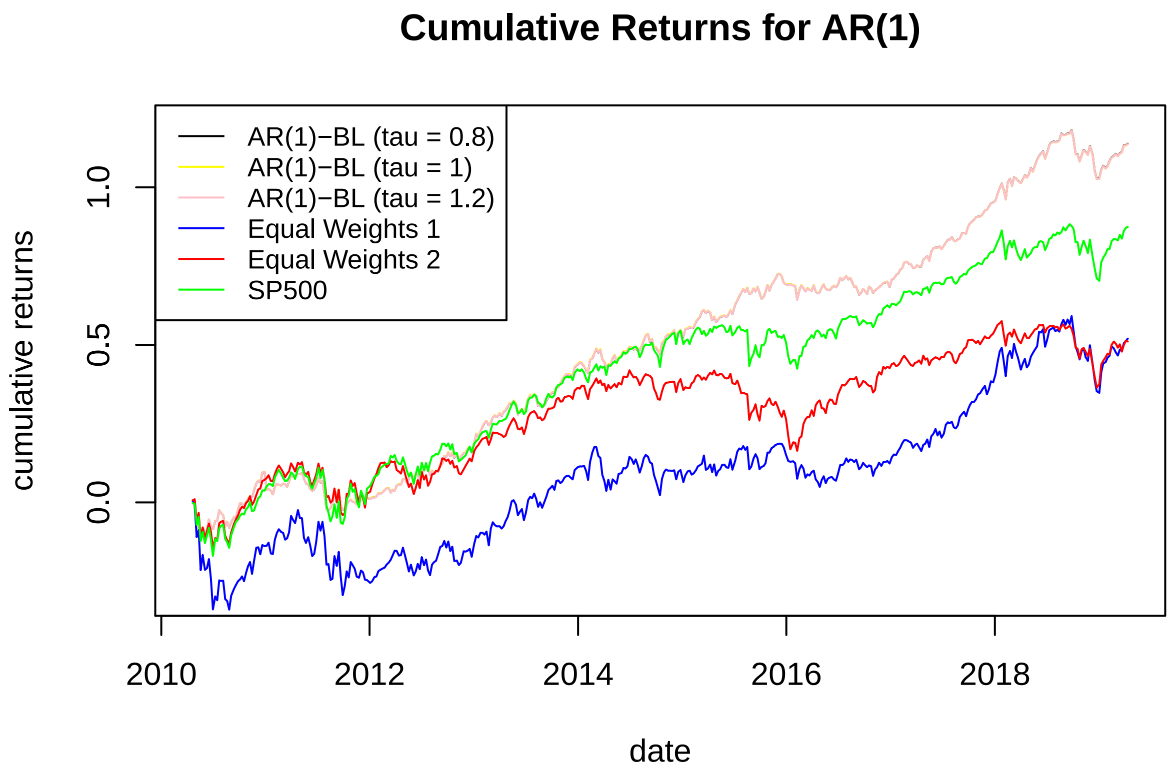
We reformat weights matrices to have dates as row names so that we can calculate turnover later.

```
weights.all1 <- weights.all1[-1,]
weights.all2 <- weights.all2[-1,]
weights.all3 <- weights.all3[-1,]
weights.all1.var <- weights.all1.var[-1,]
weights.all2.var <- weights.all2.var[-1,]
weights.all3.var <- weights.all3.var[-1,]


weights.eq1 <- weights.all1
weights.eq1[!is.na(weights.eq1)] <- 1/3174
weights.eq1[is.na(weights.eq1)] <- 0

weights.eq2 <- matrix(rep(1/3174, 1437822), nrow = 453, ncol = 3174)

weights.all1[is.na(weights.all1)] <- 0
weights.all2[is.na(weights.all2)] <- 0
weights.all3[is.na(weights.all3)] <- 0
weights.all1.var[is.na(weights.all1.var)] <- 0
weights.all2.var[is.na(weights.all2.var)] <- 0
weights.all3.var[is.na(weights.all3.var)] <- 0

rownames(weights.all1) <- dates.rw[[1]]
rownames(weights.all2) <- dates.rw[[1]]
rownames(weights.all3) <- dates.rw[[1]]
rownames(weights.all1.var) <- dates.rw[[1]]
rownames(weights.all2.var) <- dates.rw[[1]]
rownames(weights.all3.var) <- dates.rw[[1]]
rownames(weights.eq1) <- dates.rw[[1]]
rownames(weights.eq2) <- dates.rw[[1]]
```

(2)

We first plot cumulative return plot for the AR(1)-BL models in comparison to benchmarks. We also plot the most commonly chosen stocks.

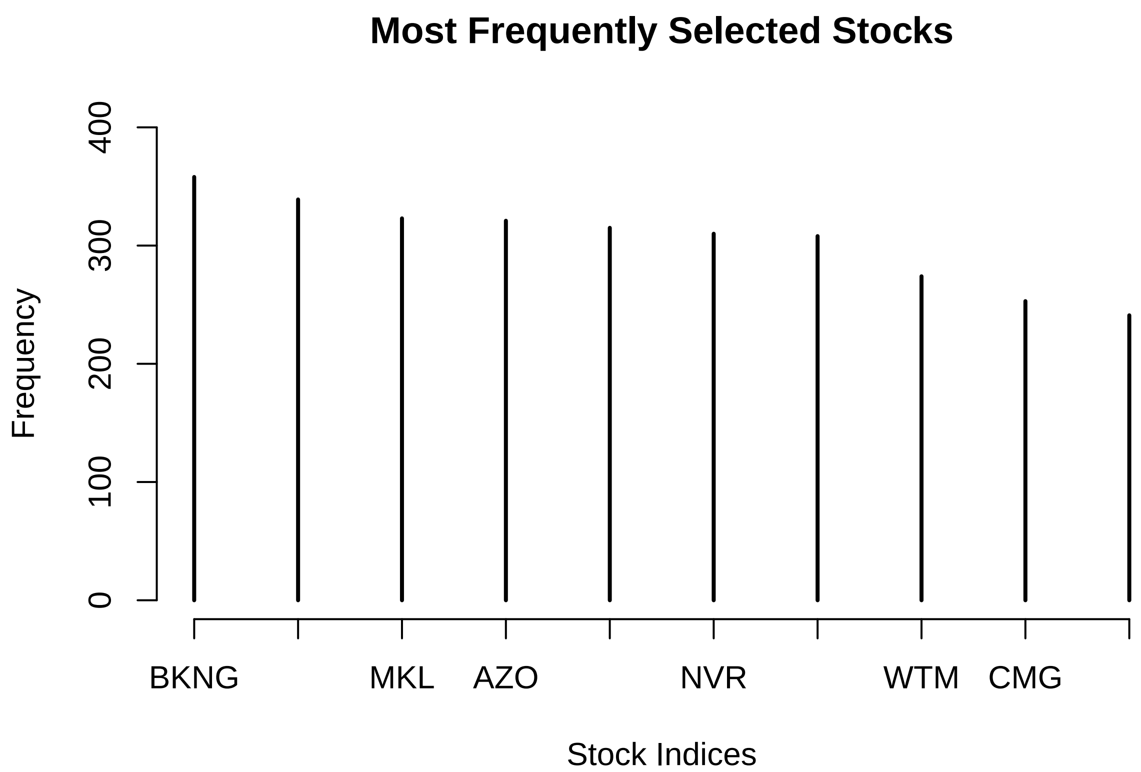
```
# Plot cumulative returns of the portfolio
plot(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.opt1), type = "l", xlab = "date", ylab = "cumulative
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.opt2), col = "yellow")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.opt3), col = "pink")

lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq1), col = "blue")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq2), col = "red")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.SP500), col = "green")
legend("topleft", legend = c("AR(1)-BL (tau = 0.8)", "AR(1)-BL (tau = 1)", "AR(1)-BL (tau = 1.2)", "Equa
```




```
# Most frequent stock indices throughout 453 iterations:
head(sort(table(names.mat), decreasing = TRUE), 30)
```

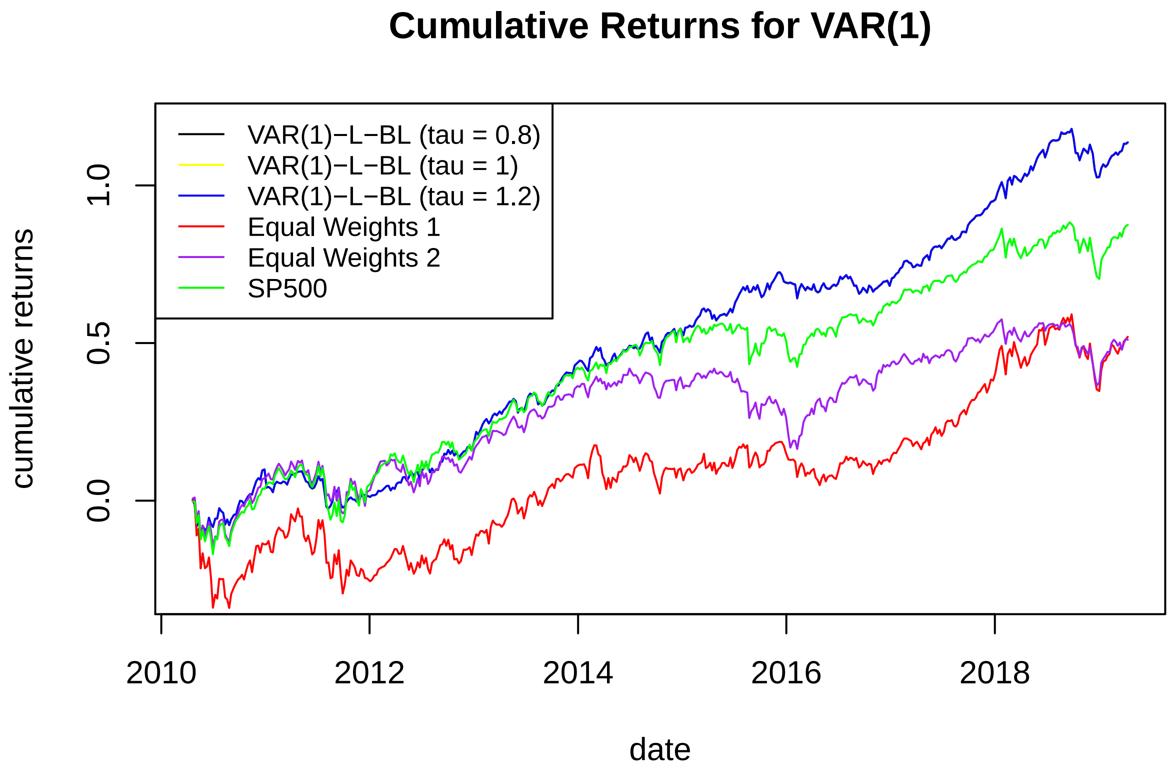
```
## names.mat
##  BKNG  AMZN   MKL   AZO GOOGL   NVR  GOOG   WTM   CMG  REGN   MTD   NEU
##   358   339   323   321   315   310   308   274   253   241   214   206
##  ISRG     Y  BIIB   GHC  EQIX  UHAL   TPL   SAM   SHW  ULTA  BIDU   ALX
##   198   192   186   184   181   171   161   157   157   153   151   150
##   BLK  ICON  MDGL  ORLY   WLL  CSGP
##   146   142   135   135   123   122
```

```
plot(sort(table(names.mat), decreasing = TRUE)[1:10], ylim = c(0, 400), main = "Most Frequently Selecte
```



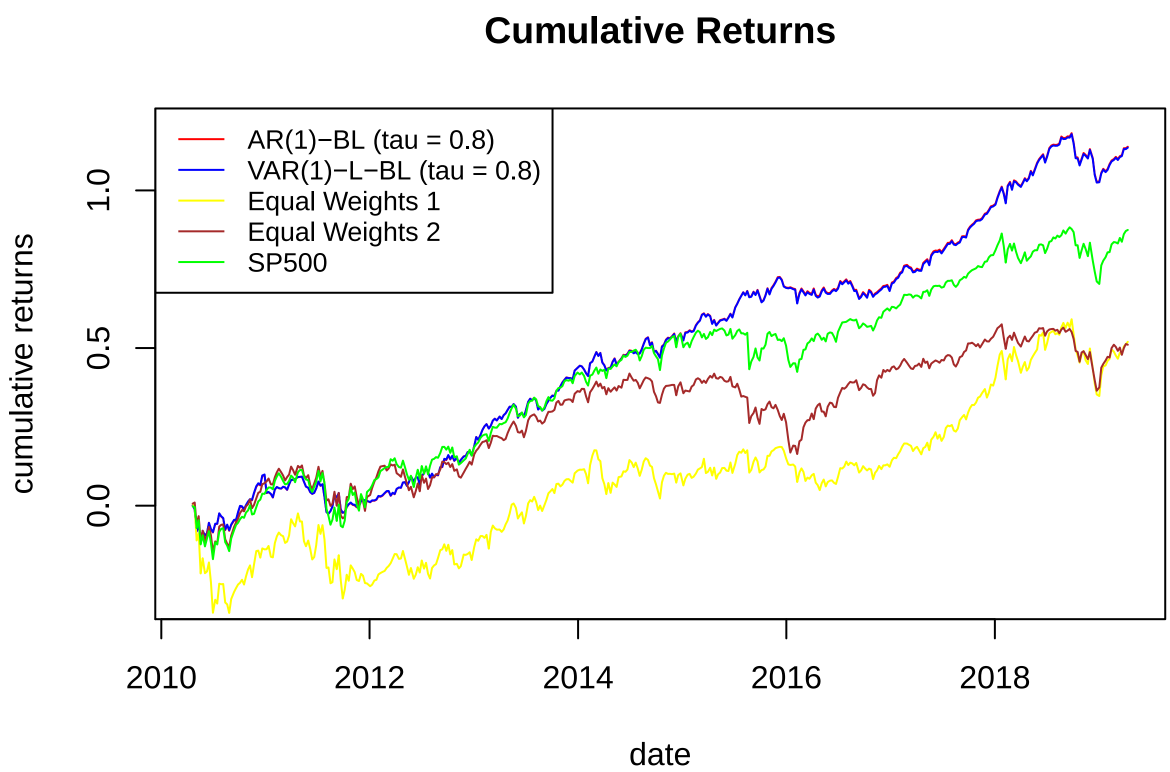
Now we plot cumulative return plot for the VAR(1)-LASSO-BL models in comparison to benchmarks.

```
plot(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns1.var), xlab = "date", ylab = "cumulative returns", ty
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns2.var), col = "yellow")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns3.var), col = "blue")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eqvar), col = "red")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq2), col = "purple")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.SP500), col = "green")
legend("topleft", legend = c("VAR(1)-L-BL (tau = 0.8)", "VAR(1)-L-BL (tau = 1)", "VAR(1)-L-BL (tau = 1.
```



Now we combine the two plots above for tau = 0.8.

```
# All cumulative plots for tau = 1
plot(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.opt1), type = "l", xlab = "date", ylab = "cumulative
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns1.var), col = "blue")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eqvar), col = "yellow")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq2), col = "brown")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.SP500), col = "green")
legend("topleft", legend = c("AR(1)-BL (tau = 0.8)", "VAR(1)-L-BL (tau = 0.8)", "Equal Weights 1", "Equa
```



(3)

We calculate mean return, volatility, Sharpe Ratio, Sortino Ratio, Maximum Drawdown and portfolio turnover for all the models.

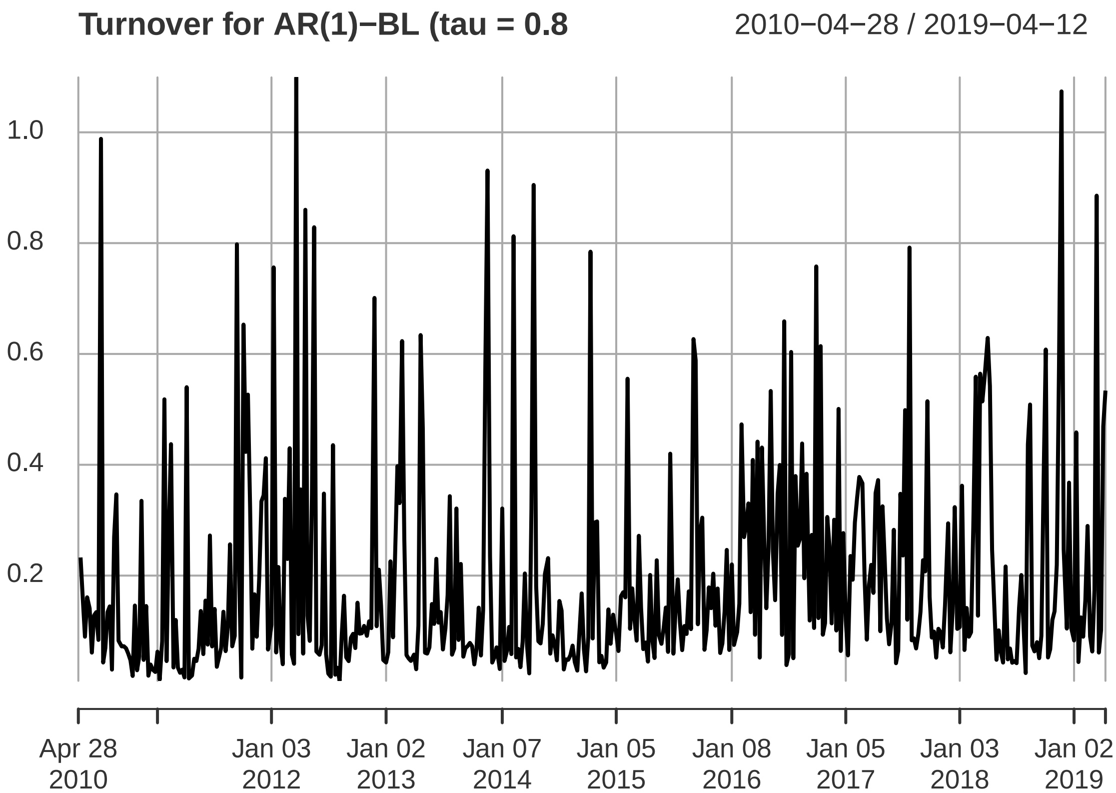
```
# Mean return
m1 <- 50*mean(returns.opt1)
m1.2 <- 50*mean(returns.opt2)
m1.3 <- 50*mean(returns.opt3)
m2.1 <- 50*mean(returns1.var)
m2.2 <- 50*mean(returns2.var)
m2.3 <- 50*mean(returns3.var)
m2 <- 50*mean(returns.eq1)
m3 <- 50*mean(returns.eq2)
m4 <- 50*mean(returns.SP500)
# Volatility
s1 <- sqrt(50)*sd(returns.opt1)
s1.2 <- sqrt(50)*sd(returns.opt2)
s1.3 <- sqrt(50)*sd(returns.opt3)
s2.1 <- sqrt(50)*sd(returns1.var)
s2.2 <- sqrt(50)*sd(returns2.var)
s2.3 <- sqrt(50)*sd(returns3.var)
s2 <- sqrt(50)*sd(returns.eq1)
s3 <- sqrt(50)*sd(returns.eq2)
s4 <- sqrt(50)*sd(returns.SP500)
# Sharpe Ratio
sr1 <- m1/s1
sr1.2 <- m1.2/s1.2
```

```r
sr1.3 <- m1.3/s1.3
sr2.1 <- m2.1/s2.1
sr2.2 <- m2.2/s2.2
sr2.3 <- m2.3/s2.3
sr2 <- m2/s2
sr3 <- m3/s3
sr4 <- m4/s4
# Sortino Ratio
sr11 <- (50/sqrt(50))*SortinoRatio(returns.opt1)
sr11.2 <- (50/sqrt(50))*SortinoRatio(returns.opt2)
sr11.3 <- (50/sqrt(50))*SortinoRatio(returns.opt3)
sr12.1 <- (50/sqrt(50))*SortinoRatio(returns1.var)
sr12.2 <- (50/sqrt(50))*SortinoRatio(returns2.var)
sr12.3 <- (50/sqrt(50))*SortinoRatio(returns3.var)
sr12 <- (50/sqrt(50))*SortinoRatio(returns.eq1)
sr13 <- (50/sqrt(50))*SortinoRatio(returns.eq2)
sr14 <- (50/sqrt(50))*SortinoRatio(returns.SP500)
# Maximum Drawdown
md1 <- maxDrawdown(returns.opt1)
md1.2 <- maxDrawdown(returns.opt2)
md1.3 <- maxDrawdown(returns.opt3)
md2.1 <- maxDrawdown(returns1.var)
md2.2 <- maxDrawdown(returns2.var)
md2.3 <- maxDrawdown(returns3.var)
md2 <- maxDrawdown(returns.eq1)
md3 <- maxDrawdown(returns.eq2)
md4 <- maxDrawdown(returns.SP500)

# Turnover
rownames(returns.cleaned) <- c(stocks.full[2:2521,1])[[1]]
returns5 <- returns.cleaned[252:2520,]
returns5 <- returns5[seq(5, nrow(returns5), 5),]

rownames(SP500.ret) <- c(stocks.full[2:2521,1])[[1]]
returns5.sp <- data.frame(SP500.ret[252:2520,])
returns5.sp <- data.frame(returns5.sp[seq(5, nrow(returns5.sp), 5),])
rownames(returns5.sp) <- rownames(returns5)


out1 <- Return.portfolio(R = returns5, weights = weights.all1, verbose = TRUE)
out2 <- Return.portfolio(R = returns5, weights = weights.all2, verbose = TRUE)
out3 <- Return.portfolio(R = returns5, weights = weights.all3, verbose = TRUE)
out1.1 <- Return.portfolio(R = returns5, weights = weights.all1.var, verbose = TRUE)
out1.2 <- Return.portfolio(R = returns5, weights = weights.all2.var, verbose = TRUE)
out1.3 <- Return.portfolio(R = returns5, weights = weights.all3.var, verbose = TRUE)
out4 <- Return.portfolio(R = returns5, weights = weights.eq1, verbose = TRUE)
out5 <- Return.portfolio(R = returns5, weights = weights.eq2, verbose = TRUE)
out6 <- Return.portfolio(R = returns5.sp, verbose = TRUE)

beginWeights1 <- out1$BOP.Weight
endWeights1 <- out1$EOP.Weight
txns1 <- beginWeights1 - lag(endWeights1)
TO1 <- xts(rowSums(abs(txns1)), order.by=index(txns1))
```

```
beginWeights2 <- out2$BOP.Weight
endWeights2 <- out2$EOP.Weight
txns2 <- beginWeights2 - lag(endWeights2)
TO2 <- xts(rowSums(abs(txns2)), order.by=index(txns2))

beginWeights3 <- out3$BOP.Weight
endWeights3 <- out3$EOP.Weight
txns3 <- beginWeights3 - lag(endWeights3)
TO3 <- xts(rowSums(abs(txns3)), order.by=index(txns3))

beginWeights1.1 <- out1.1$BOP.Weight
endWeights1.1 <- out1.1$EOP.Weight
txns1.1 <- beginWeights1.1 - lag(endWeights1.1)
TO1.1 <- xts(rowSums(abs(txns1.1)), order.by=index(txns1.1))

beginWeights1.2 <- out1.2$BOP.Weight
endWeights1.2 <- out1.2$EOP.Weight
txns1.2 <- beginWeights1.2 - lag(endWeights1.2)
TO1.2 <- xts(rowSums(abs(txns1.2)), order.by=index(txns1.2))

beginWeights1.3 <- out1.3$BOP.Weight
endWeights1.3 <- out1.3$EOP.Weight
txns1.3 <- beginWeights1.3 - lag(endWeights1.3)
TO1.3 <- xts(rowSums(abs(txns1.3)), order.by=index(txns1.3))

beginWeights4 <- out4$BOP.Weight
endWeights4 <- out4$EOP.Weight
txns4 <- beginWeights4 - lag(endWeights4)
TO4 <- xts(rowSums(abs(txns4)), order.by=index(txns4))

beginWeights5 <- out5$BOP.Weight
endWeights5 <- out5$EOP.Weight
txns5 <- beginWeights5 - lag(endWeights5)
TO5 <- xts(rowSums(abs(txns5)), order.by=index(txns5))

beginWeights6 <- out6$BOP.Weight
endWeights6 <- out6$EOP.Weight
txns6 <- beginWeights6 - lag(endWeights6)
TO6 <- xts(rowSums(abs(txns6)), order.by=index(txns6))
```

We create the turnover plot for AR(1)-BL (tau = 0.8).

```
plot(TO1, main = "Turnover for AR(1)-BL (tau = 0.8")
```



Below is the table with all results.

```r
# Results Table
data.frame(Returns=c("AR(1)-BL (tau = 0.8)", "AR(1)-BL (tau = 1)", "AR(1)-BL (tau = 1.2)", "VAR(1)-L-BL
```

```
##                   Returns       Mean Volatility Sharpe_Ratio Sortino_Ratio
## 1    AR(1)-BL (tau = 0.8) 0.12566343  0.1086669    1.1564092     1.6550115
## 2      AR(1)-BL (tau = 1) 0.12553232  0.1086733    1.1551348     1.6532093
## 3    AR(1)-BL (tau = 1.2) 0.12552235  0.1087354    1.1543840     1.6517204
## 4 VAR(1)-L-BL (tau = 0.8) 0.12535300  0.1086811    1.1534016     1.6499279
## 5   VAR(1)-L-BL (tau = 1) 0.12543254  0.1086971    1.1539635     1.6507435
## 6 VAR(1)-L-BL (tau = 1.2) 0.12548366  0.1087008    1.1543950     1.6518333
## 7         Equal Weights 1 0.05737919  0.1801309    0.3185417     0.4269687
## 8         Equal Weights 2 0.05634031  0.1383906    0.4071108     0.5584787
## 9                   SP500 0.09653791  0.1432823    0.6737601     0.9199459
##   MaxDrawdown  Turnover
## 1   0.1470579 9.5776748
## 2   0.1469537 9.5752266
## 3   0.1469777 9.5777921
## 4   0.1464400 9.5609337
## 5   0.1464840 9.5579815
## 6   0.1463828 9.5656533
## 7   0.3134621 4.6690513
## 8   0.2374286 0.5807868
## 9   0.1767571 0.0000000
```

(4)

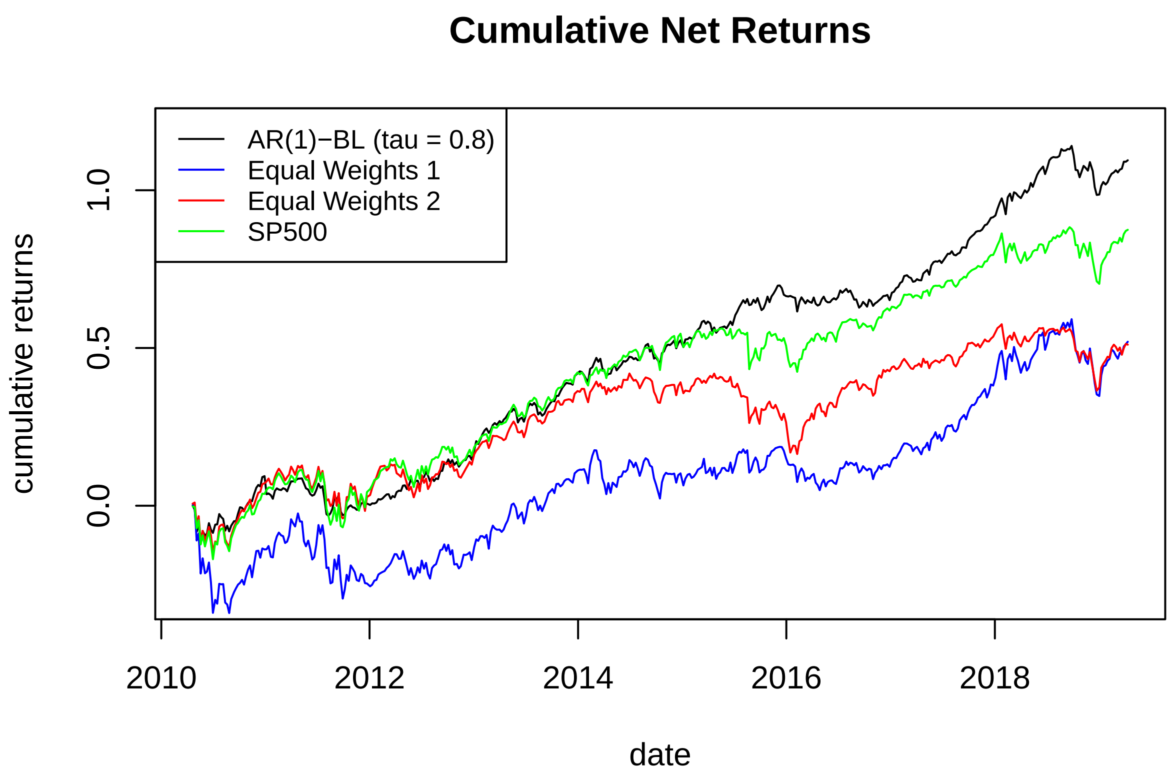
To calculate the net returns, we will take AR(1)-BL model since it seems to be the best one according to the results table from (3).

```
net_ret <- (1-0.0005*mean(TO1, na.rm = TRUE))*(1+returns.opt1)-1
m5 <- 50*mean(net_ret)
s5 <- sqrt(50)*sd(net_ret)
sr5 <- m5/s5
sr15 <- SortinoRatio(net_ret)
md5 <- maxDrawdown(net_ret)
table2 <- data.frame(Net_Returns = c("AR(1)-BL (tau = 0.8)"), Mean = m5, Volatility = s5, Sharpe_Ratio =
rownames(table2) <- NULL
table2
```

```
##            Net_Returns      Mean Volatility Sharpe_Ratio Sortino_Ratio
## 1 AR(1)-BL (tau = 0.8) 0.1208626  0.1086565     1.112336     0.2242675
##   MaxDrawdown Turnover
## 1   0.1480377 9.577675
```

We plot the cumulative returns with the net returns.

```
plot(as.Date(c(dates.rw[,1])[[1]]), cumsum(net_ret), type = "l", main = "Cumulative Net Returns", ylab =
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq1), col = "blue")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.eq2), col = "red")
lines(as.Date(c(dates.rw[,1])[[1]]), cumsum(returns.SP500), col = "green")
legend("topleft", legend=c("AR(1)-BL (tau = 0.8)", "Equal Weights 1", "Equal Weights 2", "SP500"), col=
```



Concluding, AR(1) with Black-Litterman and tau = 0.8 seems to be the best model, whose net cumulative returns outdid those of SP500 by the end of the 10 year period. It has the highest mean return, lowest volatility, highest Sharpe and Sortino ratios.