Appendix

Table A.1: Sample Periods

Sample	Formation Period (D/M/Y)	Points	Trading Period (D/M/Y)	Points	Stocks
1	6/6/2008-5/9/2009	315	6/9/2009-5/3/2010	123	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, AAA, ECM
2	6/12/2008-5/3/2010	309	6/3/2010-5/9/2010	127	
3	6/6/2009-5/9/2010	314	6/9/2010-5/3/2011	125	
4	6/12/2009-5/3/2011	313	6/3/2011-5/9/2011	128	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, AAA
5	6/6/2010-5/9/2011	316	6/9/2011-5/3/2012	126	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, AAA, Kit Digital
6	6/12/2010-5/3/2012	317	6/3/2012-5/9/2012	127	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, AAA, Kit Digital, Fortuna
7	6/6/2011-5/9/2012	317	6/9/2012-5/3/2013	123	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, AAA, Fortuna
8	6/12/2011-5/3/2013	314	6/3/2013-5/9/2013	127	Erste, KB, CEZ, Telefonica, Unipetrol, CETV, Orco, Philip Morris, Pegas, NWR, VIG, Fortuna
9	6/6/2012-5/3/2014	314	6/9/2013-5/3/2014	123	

Table A.2: Formation and Trading Statistics

Period	Stocks	Pairs	Stationary	Cointegrated	Trades, COIN	Trades, DIST
1	13	78	0	5	3	4
2	13	78	1	9	6	6
3	13	78	0	1	1	7
4	12	66	0	1	2	6
5	13	78	0	2	4	8
6	14	91	0	9	8	3
7	13	78	1	2	3	7
8	12	66	1	1	1	5
9	12	66	1	0	-	6

Stocks = original number of stocks, without detractions due to stationarity; Pairs = original number of pairs, without detractions due to stationarity or $\beta < 0$; Stationary = stocks with unit-root rejection; Cointegrated = cointegrated pairs with positive beta; Trades, COIN = number of trades generated using the cointegration method; Trades, DIST = number of trades generated using the distance method

Figure A.1: Histogram of Cointegration-Trading Returns: One-Day Lag

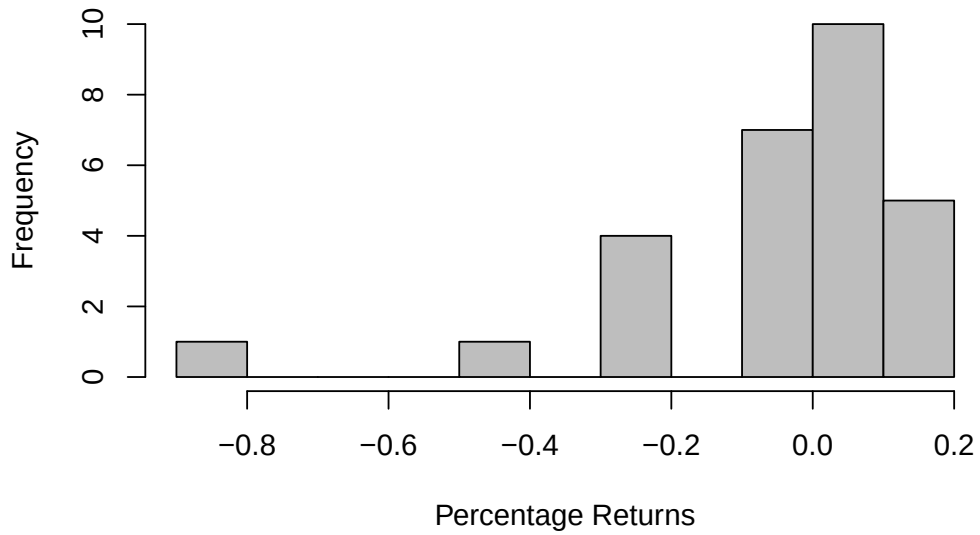

Table A.3: Return Distribution for Cointegration Trading: One-Day Lag vs. End-of-the-Day Execution

	One Day Waiting Rule	No Waiting
Average excess return*	-0.0517	-0.0527
Standard deviation	0.218	0.221
Average information ratio	0.0697	0.0607
Excess return distribution		
Median	0.0153	-0.004
Skewness	-2.579	-2.390
Excess kurtosis	8.023	6.975
Minimum	-0.896	-0.888
Maximum	0.135	0.141

* Per trade in a 6-month trading period

Figure A.2: Histogram of Distance-Trading Returns: Value-Weighted Approach, One-Day Lag

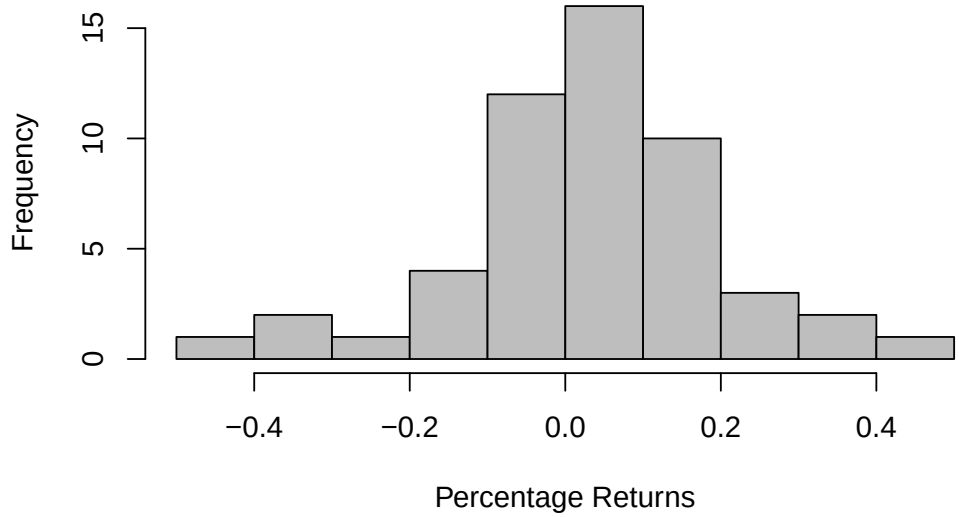

Table A.4: Return Distribution for Distance Trading: One-Day Lag vs. End-of-the-Day Execution; Value-Weighted Approach

	One Day Waiting Rule	No Waiting
Average excess return*	0.0293	0.0344
Standard deviation	0.165	0.159
Standard error	0.023	0.022
t-statistic	1.277	1.565
Average information ratio	0.1051	0.0875
Excess return distribution		
Median	0.0385	0.0241
Skewness	-0.344	-0.512
Excess kurtosis	1.439	1.055
Minimum	-0.409	-0.393
Maximum	0.431	0.368

* Per trade in a 6-month trading period

Figure A.3: Comparison of Return Histograms: One-Day Lag

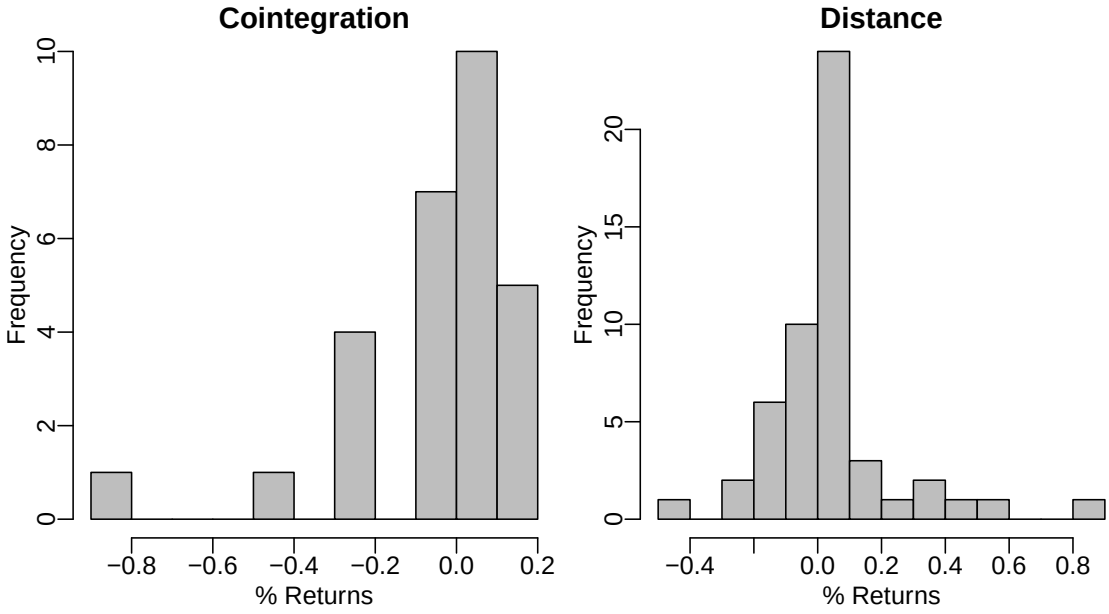

Table A.5: Comparison of Return Distributions for Cointegration and Distance Methods: One-Day Waiting Rule

	Cointegration	Distance
Average excess return	-0.0517	0.0372
Standard deviation	0.218	0.210
Average information ratio	0.0697	-0.0897
Excess return distribution		
Median	0.0153	0.0313
Skewness	-2.579	1.328
Excess kurtosis	8.023	5.114
Minimum	-0.896	-0.472
Maximum	0.135	0.843
Share of negative observations	46.4%	36.5%

Table A.6: Comparison of Distance Method with Analysis by Gatev et al. (2006)

		S&P500-Market Analysis by Gatev et al. (2006)*
Average number of pairs traded per 6-month period	4.56	4.81
Average time pairs are open in months	2.59	3.75
Average number of trades per pair	1.16	2.02
Average number of months per trade	2.23	1.86
t statistic	1.277	6.26
Excess return distribution		
Skewness	-0.344	0.34
Excess kurtosis	1.439	10.64
Minimum	-0.409	-0.126
Maximum	0.431	0.144
Observations with excess return <0	38%	35%

*measures for top 5 pairs

```
IS=lapply( paste("IS", 1:9, sep=''), get )
OOS=lapply( paste("OOS", 1:9, sep=''), get )

### COINTEGRATION METHOD ###

# FORMATION
# Unit root test for stationarity
mylist.names=c("ADF1", "ADF2", "ADF3", "ADF4", "ADF5", "ADF6", "ADF7", "ADF8", "ADF9")
ADF=sapply(mylist.names,function(x) NULL)
for(k in 1:length(IS)){
  for(i in 1:(ncol(IS[[k]])-1)){
    AA=ur.df(IS[[k]][,i+1], type=c("trend"), lags=5, selectlags=c("AIC"))
    ADF[[k]][i]=attributes(AA)$teststat[1]
  }
}

max.len=max(sapply(ADF, length))
corrected.list=lapply(ADF, function(x) {c(x, rep(0, max.len - length(x)))})
ADF=do.call(rbind, corrected.list)

# Removing stock for which unit root hypothesis was rejected
for(k in 1:nrow(ADF)){
  for(i in 1:ncol(ADF)){
    if(ADF[k,i]<(-3.41)){
      IS[[k]][,which(ADF[k,]<(-3.41))+1]=NULL
      OOS[[k]][,which(ADF[k,]<(-3.41))+1]=NULL
    }
  }}

# Residual vectors
resid1=matrix(nrow=nrow(IS1),ncol=sum(1:(ncol(IS1)-2)))
resid2=matrix(nrow=nrow(IS2),ncol=sum(1:(ncol(IS2)-2)))
resid3=matrix(nrow=nrow(IS3),ncol=sum(1:(ncol(IS3)-2)))
resid4=matrix(nrow=nrow(IS4),ncol=sum(1:(ncol(IS4)-2)))
resid5=matrix(nrow=nrow(IS5),ncol=sum(1:(ncol(IS5)-2)))
resid6=matrix(nrow=nrow(IS6),ncol=sum(1:(ncol(IS6)-2)))
resid7=matrix(nrow=nrow(IS7),ncol=sum(1:(ncol(IS7)-2)))
resid8=matrix(nrow=nrow(IS8),ncol=sum(1:(ncol(IS8)-2)))
resid9=matrix(nrow=nrow(IS9),ncol=sum(1:(ncol(IS9)-2)))
resid=lapply( paste("resid", 1:9, sep=''), get )
mju=matrix(nrow=length(IS), ncol=sum(1:(max(sapply(IS,ncol))-2)))

for(m in 1:length(IS)){
  ind=1
  for(j in 1:(ncol(IS[[m]])-1)){
    for(i in (j+1):(ncol(IS[[m]])-1)){
      if((i!=ncol(IS[[m]]))&((i!=ncol(IS[[m]])-1)|(j!=ncol(IS[[m]])-1))){
        resid[[m]][,ind]=residuals(lm(IS[[m]][,j+1]~IS[[m]][,i+1]))
        mju[m,ind]=lm(IS[[m]][,j+1]~IS[[m]][,i+1])$coeff[1]
        ind=ind+1
      }
    }}}

# Cointegration test: AEG on residual vectors
mylist.names=c("TRIADF1", "TRIADF2", "TRIADF3", "TRIADF4", "TRIADF5", "TRIADF6", "TRIADF7", "TRIADF8",
      "TRIADF9")
TRIADF=sapply(mylist.names,function(x) NULL)
for(m in 1:length(IS)){
  for(i in 1:sum(1:(ncol(IS[[m]])-2))){
    TRI=ur.df(resid[[m]][,i], type=c("drift"), lags=5, selectlags=c("AIC"))
    TRIADF[[m]][i]=attributes(TRI)$teststat[1]
  }
}

# Finding number of cointegrated pairs
max.len=max(sapply(TRIADF, length))
```

```
corrected.list=lapply(TRIADF, function(x) {c(x, rep(0, max.len - length(x)))})
TRIADF=do.call(rbind, corrected.list)

icount=NULL
for(m in 1:length(IS)){
  icount[[m]]=0
  for(i in 1:ncol(TRIADF)){
    if(TRIADF[m,i]<(-3.37)){
      icount[[m]]=icount[[m]]+1
    }
  }}
icount

# Removing sample periods with no cointegrated pairs
ind=0
for(m in 1:length(icount)){
  if(icount[m]==0){
    IS[[m-ind]]=NULL
    OOS[[m-ind]]=NULL
    TRIADF=TRIADF[-m+ind,]
    ind=ind+1
  }
}
icount=icount[icount!=0]

# Cointegrated pairs, ordered from lowest t-statistic
ndx=NULL
for(m in 1:length(IS)){
  ndx[[m]]=order(TRIADF[m,])[1:icount[[m]]]
}

# TRADING
# Cointegration coefficient (ratio in which to trade stocks)
beta=matrix(nrow=length(IS), ncol=sum(1:(max(sapply(IS,ncol))-2)))
for(m in 1:length(IS)){
  ind=1
  for(j in 1:(ncol(IS[[m]])-1)){
    for(i in (j+1):(ncol(IS[[m]])-1)){
      if((i!=ncol(IS[[m]]))&((i!=ncol(IS[[m]])-1)|(j!=ncol(IS[[m]])-1))){
        beta[m,ind]=lm(IS[[m]][,j+1]~IS[[m]][,i+1])$coeff[2]
        ind=ind+1
      }
    }}}

# Formation of trading-period spreads
regoos1=matrix(nrow=nrow(OOS[[1]]),ncol=sum(1:(ncol(OOS[[1]])-2)))
regoos2=matrix(nrow=nrow(OOS[[2]]),ncol=sum(1:(ncol(OOS[[2]])-2)))
regoos3=matrix(nrow=nrow(OOS[[3]]),ncol=sum(1:(ncol(OOS[[3]])-2)))
regoos4=matrix(nrow=nrow(OOS[[4]]),ncol=sum(1:(ncol(OOS[[4]])-2)))
regoos5=matrix(nrow=nrow(OOS[[5]]),ncol=sum(1:(ncol(OOS[[5]])-2)))
regoos6=matrix(nrow=nrow(OOS[[6]]),ncol=sum(1:(ncol(OOS[[6]])-2)))
regoos7=matrix(nrow=nrow(OOS[[7]]),ncol=sum(1:(ncol(OOS[[7]])-2)))
regoos8=matrix(nrow=nrow(OOS[[8]]),ncol=sum(1:(ncol(OOS[[8]])-2)))
regoos=lapply( paste("regoos", 1:8, sep=''), get )

for(m in 1:length(regoos)){
  ind=1
  for(j in 1:(ncol(OOS[[m]])-1)){
    for(i in (j+1):(ncol(OOS[[m]])-1)){
      if((i!=ncol(OOS[[m]]))&((i!=ncol(OOS[[m]])-1)|(j!=ncol(OOS[[m]])-1))){
        regoos[[m]][,ind]=OOS[[m]][,j+1]-beta[m,ind]*OOS[[m]][,i+1]
        ind=ind+1
      }
    }}
  regoos[[m]]=regoos[[m]][,ndx[[m]]]
  regoos[[m]]=as.matrix(regoos[[m]])}

```

```
# Means and standard deviations
model1=matrix(nrow=nrow(IS1),ncol=sum(1:(ncol(IS1)-2)))
model2=matrix(nrow=nrow(IS2),ncol=sum(1:(ncol(IS2)-2)))
model3=matrix(nrow=nrow(IS3),ncol=sum(1:(ncol(IS3)-2)))
model4=matrix(nrow=nrow(IS4),ncol=sum(1:(ncol(IS4)-2)))
model5=matrix(nrow=nrow(IS5),ncol=sum(1:(ncol(IS5)-2)))
model6=matrix(nrow=nrow(IS6),ncol=sum(1:(ncol(IS6)-2)))
model7=matrix(nrow=nrow(IS7),ncol=sum(1:(ncol(IS7)-2)))
model8=matrix(nrow=nrow(IS8),ncol=sum(1:(ncol(IS8)-2)))
model=lapply(paste("model", 1:8, sep=''), get)

for(m in 1:length(model)){
  ind=1
  for(j in 1:sum(1:(ncol(IS[[m]])-2))){
    for(k in 1:nrow(resid[[m]])){
      model[[m]][k,j]=resid[[m]][k,j] +mju[m,j]
      ind=ind+1
    }}
  model[[m]]=model[[m]][,ndx[[m]]]
  model[[m]]=as.matrix(model[[m]])}

mylist.names=c("mean1", "mean2", "mean3", "mean4", "mean5", "mean6", "mean7", "mean8")
mean=sapply(mylist.names,function(x) NULL)
mylist.names=c("sd1", "sd2", "sd3", "sd4", "sd5", "sd6", "sd7", "sd8")
sd=sapply(mylist.names,function(x) NULL)

for(m in 1:length(model)){
  mean[[m]]=colMeans(model[[m]])
  sd[[m]]=colStdevs(model[[m]])
}

max.len=max(sapply(mean, length))
corrected.list=lapply(mean, function(x) {c(x, rep(0, max.len - length(x)))})
mean=do.call(rbind, corrected.list)
max.len=max(sapply(sd, length))
corrected.list=lapply(sd, function(x) {c(x, rep(0, max.len - length(x)))})
sd=do.call(rbind, corrected.list)

# Trading signals
a=c(1,1,1,1)
b=c(-2,0,2,0)
Thresholds=array(rep(NA), dim=c(nrow(mean),ncol(mean),length(a)))
for(k in 1:length(a)){
  for(i in 1:nrow(mean)){
    for(j in 1:ncol(mean)){
      Thresholds[i,j,k]=a[k]*mean[i,j]+b[k]*sd[i,j]
    }
  }}

# Points of entry/exit: finding rows where spreads are below or above trading signals
iakcie=NULL
rtime=NULL
for(m in 1:length(regoos)){
  iakcie[m]=ncol(regoos[[m]]); rtime[m]=nrow(regoos[[m]])
}
indBuyL=NULL; indSellL=NULL; indSellS=NULL; indBuyS=NULL

for(m in 1:length(regoos)){
  indBuyL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  indBuyL[[m]][is.na(indBuyL[[m]])]=0
  indSellL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  indSellL[[m]][is.na(indSellL[[m]])]=0
  indSellS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  indSellS[[m]][is.na(indSellS[[m]])]=0
  indBuyS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  indBuyS[[m]][is.na(indBuyS[[m]])]=0
}
```

```
200
201 for(m in 1:length(regoos)){
202   for(i in 1:iakcie[m]){
203     ind1=1;ind2=1;ind3=1;ind4=1
204     for(j in 1:rtime[m]){
205       if(regoos[[m]][j,i]<=Thresholds[m,i,1]){indBuyL[[m]][ind1,i]=j
206                                               ind1=ind1+1}
207       if(regoos[[m]][j,i]>=Thresholds[m,i,2]) {indSellL[[m]][ind2,i]=j
208                                                ind2=ind2+1}
209       if(regoos[[m]][j,i]>=Thresholds[m,i,3]) {indSellS[[m]][ind3,i]=j
210                                                ind3=ind3+1}
211       if(regoos[[m]][j,i]<=Thresholds[m,i,4]) {indBuyS[[m]][ind4,i]=j
212                                                ind4=ind4+1}
213     }
214   }}
215
216 ### Calculating profits
217 ## In each time
218 # Long and short spreads
219 profitbottomup1=matrix(nrow=nrow(OOS[[1]])-1,ncol=ncol(regoos[[1]]))
220 profitbottomup2=matrix(nrow=nrow(OOS[[2]])-1,ncol=ncol(regoos[[2]]))
221 profitbottomup3=matrix(nrow=nrow(OOS[[3]])-1,ncol=ncol(regoos[[3]]))
222 profitbottomup4=matrix(nrow=nrow(OOS[[4]])-1,ncol=ncol(regoos[[4]]))
223 profitbottomup5=matrix(nrow=nrow(OOS[[5]])-1,ncol=ncol(regoos[[5]]))
224 profitbottomup6=matrix(nrow=nrow(OOS[[6]])-1,ncol=ncol(regoos[[6]]))
225 profitbottomup7=matrix(nrow=nrow(OOS[[7]])-1,ncol=ncol(regoos[[7]]))
226 profitbottomup8=matrix(nrow=nrow(OOS[[8]])-1,ncol=ncol(regoos[[8]]))
227 profitbottomup=lapply( paste("profitbottomup", 1:8, sep=''), get )
228
229 profitupbottom1=matrix(nrow=nrow(OOS[[1]])-1,ncol=ncol(regoos[[1]]))
230 profitupbottom2=matrix(nrow=nrow(OOS[[2]])-1,ncol=ncol(regoos[[2]]))
231 profitupbottom3=matrix(nrow=nrow(OOS[[3]])-1,ncol=ncol(regoos[[3]]))
232 profitupbottom4=matrix(nrow=nrow(OOS[[4]])-1,ncol=ncol(regoos[[4]]))
233 profitupbottom5=matrix(nrow=nrow(OOS[[5]])-1,ncol=ncol(regoos[[5]]))
234 profitupbottom6=matrix(nrow=nrow(OOS[[6]])-1,ncol=ncol(regoos[[6]]))
235 profitupbottom7=matrix(nrow=nrow(OOS[[7]])-1,ncol=ncol(regoos[[7]]))
236 profitupbottom8=matrix(nrow=nrow(OOS[[8]])-1,ncol=ncol(regoos[[8]]))
237 profitupbottom=lapply( paste("profitupbottom", 1:8, sep=''), get )
238
239 for(m in 1:length(profitbottomup)){
240   for(i in 1:ncol(regoos[[m]])){
241     profitbottomup[[m]][,i]=diff(regoos[[m]][,i])
242     profitupbottom[[m]][,i]=-diff(regoos[[m]][,i])
243   }
244   null=rep(0, length=ncol(regoos[[m]]))
245   profitbottomup[[m]]=rbind(null, profitbottomup[[m]])
246   profitupbottom[[m]]=rbind(null, profitupbottom[[m]])
247 }
248
249
250 ## For each particular trade
251 PXOOS=lapply( paste("PXOOS", 1:length(IS), sep=''), get )
252 # Long
253 buyL=NULL
254 sellL=NULL
255 profitL=NULL
256 profitPXL=NULL
257 for(m in 1:length(regoos)){
258   buyL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
259   sellL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
260   profitL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
261   profitPXL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
262 }
263
264 for(m in 1:length(regoos)){
265   for(i in 1:iakcie[m]){
266     ind=1
```

```
    isellL=0
    for(j in 1:rtime[m]){
      if(indBuyL[[m]][j,i]>isellL){
        for(k in 1:rtime[m]){
          if((indSellL[[m]][k,i]>=indBuyL[[m]][j,i])&(indBuyL[[m]][j,i]!=rtime[m])){
            ibuyL=indBuyL[[m]][j,i]+1
            isellL=indSellL[[m]][k,i]+1
            buyL[[m]][ind,i]=indBuyL[[m]][j,i]+1
            sellL[[m]][ind,i]=indSellL[[m]][k,i]+1
            profitL[[m]][ind,i]=sum(profitbottomup[[m]][(buyL[[m]][ind,i]+1):sellL[[m]][ind,i],i])
            profitPXL[[m]][ind,i]=sum(PXOOS[[m]][(buyL[[m]][ind,i]+1):sellL[[m]][ind,i],3])
            ind=ind+1
            break
          }
          if((k==rtime[m])&(indBuyL[[m]][j,i]!=rtime[m])){
            ibuyL=indBuyL[[m]][j,i]+1
            isellL=rtime[m]
            buyL[[m]][ind,i]=indBuyL[[m]][j,i]+1
            sellL[[m]][ind,i]=rtime[m]
            profitL[[m]][ind,i]=sum(profitbottomup[[m]][(buyL[[m]][ind,i]+1):sellL[[m]][ind,i],i])
            profitPXL[[m]][ind,i]=sum(PXOOS[[m]][(buyL[[m]][ind,i]+1):sellL[[m]][ind,i],3])
            ind=ind+1
          }
        }
      }}}
  buyL[[m]][is.na(buyL[[m]])]=0
  sellL[[m]][is.na(sellL[[m]])]=0
  buyL[[m]]=as.matrix(buyL[[m]])
  sellL[[m]]=as.matrix(sellL[[m]])
}

# Information ratio, long
IRL=array(rep(NA), dim=c(4,max(sapply(buyL, ncol)),length(buyL)))
for(m in 1:length(regoos)){
  for(i in 1:ncol(buyL[[m]])){
    for(k in 1:nrow(buyL[[m]])){
      if(buyL[[m]][k,i]!=0){
        IRL[k,i,m]=mean(profitbottomup[[m]][(buyL[[m]][k,i]+1):sellL[[m]][k,i],i]-PXOOS[[m]][(buyL[[m
            ]][k,i]+1):sellL[[m]][k,i],3])/sd(profitbottomup[[m]][(buyL[[m]][k,i]+1):sellL[[m]][k,i],
            i]-PXOOS[[m]][(buyL[[m]][k,i]+1):sellL[[m]][k,i],3])
      }
    }}}
IRL

# Short
sellS=NULL
buyS=NULL
profitS=NULL
profitPXS=NULL
for(m in 1:length(regoos)){
  sellS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  buyS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  profitS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
  profitPXS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
}

for(m in 1:length(regoos)){
  for(i in 1:iakcie[m]){
    ind=1
    ibuyS=0
    for(j in 1:rtime[m]){
      if(indSellS[[m]][j,i]>ibuyS){
        for(k in 1:rtime[m]){
          if((indBuyS[[m]][k,i]>=indSellS[[m]][j,i])&(indSellS[[m]][j,i]!=rtime[m])){
            isellS=indSellS[[m]][j,i]+1
            ibuyS=indBuyS[[m]][k,i]+1
            sellS[[m]][ind,i]=indSellS[[m]][j,i]+1
```

```
            buyS[[m]][ind,i]=indBuyS[[m]][k,i]+1
            profitS[[m]][ind,i]=sum(profitupbottom[[m]][(sellS[[m]][ind,i]+1):buyS[[m]][ind,i],i])
            profitPXS[[m]][ind,i]=sum(PXOOS[[m]][(sellS[[m]][ind,i]+1):buyS[[m]][ind,i],3])
            ind=ind+1
            break
          }
          if((k==rtime[m])&(indSellS[[m]][j,i]!=rtime[m])){
            isellS=indSellS[[m]][j,i]+1
            ibuyS=rtime[m]
            sellS[[m]][ind,i]=indSellS[[m]][j,i]+1
            buyS[[m]][ind,i]=ibuyS
            profitS[[m]][ind,i]=sum(profitupbottom[[m]][(sellS[[m]][ind,i]+1):buyS[[m]][ind,i],i])
            profitPXS[[m]][ind,i]=sum(PXOOS[[m]][(sellS[[m]][ind,i]+1):buyS[[m]][ind,i],3])
            ind=ind+1
          }
        }
      }}}
  sellS[[m]][is.na(sellS[[m]])]=0
  buyS[[m]][is.na(buyS[[m]])]=0
  sellS[[m]]=as.matrix(sellS[[m]])
  buyS[[m]]=as.matrix(buyS[[m]])
}

# Information ratio, short
IRS=array(rep(NA), dim=c(4,max(sapply(sellS, ncol)),length(sellS)))
for(m in 1:length(regoos)){
  for(i in 1:ncol(sellS[[m]])){
    for(k in 1:nrow(sellS[[m]])){
      if(sellS[[m]][k,i]!=0){
        IRS[k,i,m]=mean(profitupbottom[[m]][(sellS[[m]][k,i]+1):buyS[[m]][k,i],i]-PXOOS[[m]][(sellS[[m
            ]][k,i]+1):buyS[[m]][k,i],3])/sd(profitupbottom[[m]][(sellS[[m]][k,i]+1):buyS[[m]][k,i],i
            ]-PXOOS[[m]][(sellS[[m]][k,i]+1):buyS[[m]][k,i],3])
      }
    }}}
IRS

### DISTANCE METHOD ###

# Logarithms of prices
IS=lapply(paste("IS", 1:9, sep=''), get)
OOS=lapply(paste("OOS", 1:9, sep=''), get)

# Absolute values of prices
PIS=lapply(paste("PIS", 1:9, sep=''), get)
POOS=lapply(paste("POOS", 1:9, sep=''), get)

ISa=list(); OOSa=list(); PISa=list(); POOSa=list()
for(m in 1:length(IS)){
  ISa[[m]]=IS[[m]][,-1]
  OOSa[[m]]=OOS[[m]][,-1]
  PISa[[m]]=PIS[[m]][,-1]
  POOSa[[m]]=POOS[[m]][,-1]
}

# FORMATION
ret1=matrix(nrow=nrow(IS[[1]])-1,ncol=ncol(ISa[[1]]))
ret2=matrix(nrow=nrow(IS[[2]])-1,ncol=ncol(ISa[[2]]))
ret3=matrix(nrow=nrow(IS[[3]])-1,ncol=ncol(ISa[[3]]))
ret4=matrix(nrow=nrow(IS[[4]])-1,ncol=ncol(ISa[[4]]))
ret5=matrix(nrow=nrow(IS[[5]])-1,ncol=ncol(ISa[[5]]))
ret6=matrix(nrow=nrow(IS[[6]])-1,ncol=ncol(ISa[[6]]))
ret7=matrix(nrow=nrow(IS[[7]])-1,ncol=ncol(ISa[[7]]))
ret8=matrix(nrow=nrow(IS[[8]])-1,ncol=ncol(ISa[[8]]))
ret9=matrix(nrow=nrow(IS[[9]])-1,ncol=ncol(ISa[[9]]))
ret=lapply( paste("ret", 1:9, sep=''), get )

for(m in 1:length(ISa)){
```

```
  for(i in 1:ncol(ISa[[m]])){
    ret[[m]][,i]=diff(ISa[[m]][,i])
  }
  null=rep(0, length=ncol(ISa[[m]]))
  ret[[m]]=rbind(null, ret[[m]])
}

mylist.names=c("meanD1", "meanD2", "meanD3", "meanD4", "meanD5", "meanD6", "meanD7", "meanD8")
meanD=sapply(mylist.names,function(x) NULL)
mylist.names=c("sdD1", "sdD2", "sdD3", "sdD4", "sdD5", "sdD6", "sdD7", "sdD8")
sdD=sapply(mylist.names,function(x) NULL)

for(m in 1:length(ret)){
  meanD[[m]]=colMeans(ret[[m]])
  sdD[[m]]=colStdevs(ret[[m]])
}

# Cumulative sum of standardized returns
retN1=matrix(nrow=nrow(ret[[1]]),ncol=ncol(ret[[1]]))
retN2=matrix(nrow=nrow(ret[[2]]),ncol=ncol(ret[[2]]))
retN3=matrix(nrow=nrow(ret[[3]]),ncol=ncol(ret[[3]]))
retN4=matrix(nrow=nrow(ret[[4]]),ncol=ncol(ret[[4]]))
retN5=matrix(nrow=nrow(ret[[5]]),ncol=ncol(ret[[5]]))
retN6=matrix(nrow=nrow(ret[[6]]),ncol=ncol(ret[[6]]))
retN7=matrix(nrow=nrow(ret[[7]]),ncol=ncol(ret[[7]]))
retN8=matrix(nrow=nrow(ret[[8]]),ncol=ncol(ret[[8]]))
retN9=matrix(nrow=nrow(ret[[9]]),ncol=ncol(ret[[9]]))

retN=lapply( paste("retN", 1:9, sep=''), get )
for(m in 1:length(ret)){
  for(i in 1:ncol(ret[[m]])){
    for(k in 1:nrow(ret[[m]])){
      retN[[m]][k,i]=(ret[[m]][k,i]-meanD[[m]][i])/sdD[[m]][i]
    }
    retN[[m]][,i]=cumsum(retN[[m]][,i])
  }}

# Sum of squared deviations of normalized returns
devN=matrix(nrow=length(retN), ncol=sum(1:(max(sapply(retN, ncol))-1)))
for(m in 1:length(retN)){
  ind=1
  for(j in 1:(ncol(retN[[m]])-1)){
    for(i in (j+1):ncol(retN[[m]])){
      devN[m,ind]=sum((retN[[m]][,j]-retN[[m]][,i])^2)
      ind=ind+1
    }
  }}

# Pairs with minimized distance, ordered from the lowest SSD
bdx=NULL
for(m in 1:nrow(devN)){
  bdx[[m]]=order(devN[m,])[1:5]
  bdx[[m]]=as.matrix(bdx[[m]])
}

# TRADING
# Retrieving stocks from pairs
stock1=array(rep(0), dim=c(1,max(sapply(bdx, length)),length(ISa)))
stock2=array(rep(0), dim=c(1,max(sapply(bdx, length)),length(ISa)))
for(m in 1:length(ISa)){
  for(n in 1:length(bdx[[m]])){
    ind=1
    for(j in 1:(ncol(ISa[[m]])-1)){
      for(i in (j+1):ncol(ISa[[m]])){
        if(bdx[[m]][n]==ind){
          stock1[1,n,m]=j
          stock2[1,n,m]=i
```

```
        }
        ind=ind+1
      }
    }}}

# Standardization of OOS returns
retoos1=matrix(nrow=nrow(OOSa[[1]])-1,ncol=ncol(OOSa[[1]]))
retoos2=matrix(nrow=nrow(OOSa[[2]])-1,ncol=ncol(OOSa[[2]]))
retoos3=matrix(nrow=nrow(OOSa[[3]])-1,ncol=ncol(OOSa[[3]]))
retoos4=matrix(nrow=nrow(OOSa[[4]])-1,ncol=ncol(OOSa[[4]]))
retoos5=matrix(nrow=nrow(OOSa[[5]])-1,ncol=ncol(OOSa[[5]]))
retoos6=matrix(nrow=nrow(OOSa[[6]])-1,ncol=ncol(OOSa[[6]]))
retoos7=matrix(nrow=nrow(OOSa[[7]])-1,ncol=ncol(OOSa[[7]]))
retoos8=matrix(nrow=nrow(OOSa[[8]])-1,ncol=ncol(OOSa[[8]]))
retoos9=matrix(nrow=nrow(OOSa[[9]])-1,ncol=ncol(OOSa[[9]]))
retoos=lapply( paste("retoos", 1:9, sep=''), get )

retoosw1=matrix(nrow=nrow(OOSa[[1]])-1,ncol=ncol(OOSa[[1]]))
retoosw2=matrix(nrow=nrow(OOSa[[2]])-1,ncol=ncol(OOSa[[2]]))
retoosw3=matrix(nrow=nrow(OOSa[[3]])-1,ncol=ncol(OOSa[[3]]))
retoosw4=matrix(nrow=nrow(OOSa[[4]])-1,ncol=ncol(OOSa[[4]]))
retoosw5=matrix(nrow=nrow(OOSa[[5]])-1,ncol=ncol(OOSa[[5]]))
retoosw6=matrix(nrow=nrow(OOSa[[6]])-1,ncol=ncol(OOSa[[6]]))
retoosw7=matrix(nrow=nrow(OOSa[[7]])-1,ncol=ncol(OOSa[[7]]))
retoosw8=matrix(nrow=nrow(OOSa[[8]])-1,ncol=ncol(OOSa[[8]]))
retoosw9=matrix(nrow=nrow(OOSa[[9]])-1,ncol=ncol(OOSa[[9]]))
retoosw=lapply( paste("retoosw", 1:9, sep=''), get )

for(m in 1:length(OOSa)){
  for(i in 1:ncol(OOSa[[m]])){
    retoos[[m]][,i]=diff(OOSa[[m]][,i])
  }
  null=rep(0, length=ncol(OOSa[[m]]))
  retoos[[m]]=rbind(null, retoos[[m]])
  retoosw[[m]]=retoos[[m]]+1
}

OOSN1=matrix(nrow=nrow(OOSa[[1]]),ncol=ncol(OOSa[[1]]))
OOSN2=matrix(nrow=nrow(OOSa[[2]]),ncol=ncol(OOSa[[2]]))
OOSN3=matrix(nrow=nrow(OOSa[[3]]),ncol=ncol(OOSa[[3]]))
OOSN4=matrix(nrow=nrow(OOSa[[4]]),ncol=ncol(OOSa[[4]]))
OOSN5=matrix(nrow=nrow(OOSa[[5]]),ncol=ncol(OOSa[[5]]))
OOSN6=matrix(nrow=nrow(OOSa[[6]]),ncol=ncol(OOSa[[6]]))
OOSN7=matrix(nrow=nrow(OOSa[[7]]),ncol=ncol(OOSa[[7]]))
OOSN8=matrix(nrow=nrow(OOSa[[8]]),ncol=ncol(OOSa[[8]]))
OOSN9=matrix(nrow=nrow(OOSa[[9]]),ncol=ncol(OOSa[[9]]))
OOSN=lapply( paste("OOSN", 1:9, sep=''), get )

for(m in 1:length(OOSa)){
  for(i in 1:ncol(OOSa[[m]])){
    for(k in 1:nrow(OOSa[[m]])){
      OOSN[[m]][k,i]=(retoos[[m]][k,i]-meanD[[m]][i])/sdD[[m]][i]
    }
    OOSN[[m]][,i]=cumsum(OOSN[[m]][,i])
  }}

# For chosen pairs, difference between normalized cumulative returns
dif1=matrix(nrow=nrow(OOSN[[1]]),ncol=nrow(bdx[[1]]))
dif2=matrix(nrow=nrow(OOSN[[2]]),ncol=nrow(bdx[[2]]))
dif3=matrix(nrow=nrow(OOSN[[3]]),ncol=nrow(bdx[[3]]))
dif4=matrix(nrow=nrow(OOSN[[4]]),ncol=nrow(bdx[[4]]))
dif5=matrix(nrow=nrow(OOSN[[5]]),ncol=nrow(bdx[[5]]))
dif6=matrix(nrow=nrow(OOSN[[6]]),ncol=nrow(bdx[[6]]))
dif7=matrix(nrow=nrow(OOSN[[7]]),ncol=nrow(bdx[[7]]))
dif8=matrix(nrow=nrow(OOSN[[8]]),ncol=nrow(bdx[[8]]))
dif9=matrix(nrow=nrow(OOSN[[9]]),ncol=nrow(bdx[[9]]))
dif=lapply( paste("dif", 1:9, sep=''), get )
```

```
531
532 for(m in 1:length(OOSN)){
533   for(n in 1:length(stock1[1,,m])){
534     if(stock1[1,n,m]!=0){
535       dif[[m]][,n]=OOSN[[m]][,stock1[1,n,m]]-OOSN[[m]][,stock2[1,n,m]]
536     }
537   }
538   dif[[m]]=as.matrix(dif[[m]])
539 }
540
541 # Trading signals
542 meanN=array(rep(NA), dim=c(1,ncol(stock1),length(retN)))
543 sdN=array(rep(NA), dim=c(1,ncol(stock1),length(retN)))
544 for(m in 1:length(retN)){
545   for(n in 1:ncol(stock1)){
546     if(stock1[1,n,m]!=0){
547       meanN[1,n,m]=mean(retN[[m]][,stock1[1,n,m]]-retN[[m]][,stock2[1,n,m]])
548       sdN[1,n,m]=sd(retN[[m]][,stock1[1,n,m]]-retN[[m]][,stock2[1,n,m]])
549     }
550   }}
551
552 a=c(1,1,1,1)
553 b=c(-2,0,2,0)
554 Thresholds=array(rep(NA), dim=c(length(OOSN),ncol(meanN),length(b)))
555 for(k in 1:length(b)){
556   for(i in 1:length(OOSN)){
557     for(j in 1:ncol(meanN)){
558       Thresholds[i,j,k]=a[k]*meanN[1,j,i]+b[k]*sdN[1,j,i]
559     }
560   }}
561
562 # Points of entry/exit: finding rows where spreads are below or above trading signals
563 iakcie=NULL
564 rtime=NULL
565 for(m in 1:length(dif)){
566   iakcie[m]=ncol(dif[[m]]); rtime[m]=nrow(dif[[m]])
567 }
568 indBuyL=NULL; indSellL=NULL; indSellS=NULL; indBuyS=NULL
569
570 for(m in 1:length(dif)){
571   indBuyL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
572   indBuyL[[m]][is.na(indBuyL[[m]])]=0
573   indSellL[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
574   indSellL[[m]][is.na(indSellL[[m]])]=0
575   indSellS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
576   indSellS[[m]][is.na(indSellS[[m]])]=0
577   indBuyS[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
578   indBuyS[[m]][is.na(indBuyS[[m]])]=0
579 }
580
581 for(m in 1:length(dif)){
582   for(i in 1:iakcie[m]){
583     ind1=1;ind2=1;ind3=1;ind4=1
584     for(j in 1:rtime[m]){
585       if(dif[[m]][j,i]<=Thresholds[m,i,1]){indBuyL[[m]][ind1,i]=j
586                                            ind1=ind1+1}
587       if(dif[[m]][j,i]>=Thresholds[m,i,2]) {indSellL[[m]][ind2,i]=j
588                                            ind2=ind2+1}
589       if(dif[[m]][j,i]>=Thresholds[m,i,3]) {indSellS[[m]][ind3,i]=j
590                                            ind3=ind3+1}
591       if(dif[[m]][j,i]<=Thresholds[m,i,4]) {indBuyS[[m]][ind4,i]=j
592                                            ind4=ind4+1}
593     }
594   }}
595
596 ### Calculating profits
597 PXOOS=lapply( paste("PXOOS", 1:length(IS), sep=''), get )
```

```
598
599 ## Long ##
600 buyLD=NULL
601 sellLD=NULL
602 for(m in 1:length(dif)){
603   buyLD[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
604   sellLD[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
605 }
606
607 for(m in 1:length(dif)){
608   for(i in 1:iakcie[m]){
609     ind=1
610     isellL=0
611     for(j in 1:rtime[m]){
612       if(indBuyL[[m]][j,i]>isellL){
613         for(k in 1:rtime[m]){
614           if((indSellL[[m]][k,i]>=indBuyL[[m]][j,i])&(indBuyL[[m]][j,i]!=rtime[m])){
615             ibuyL=indBuyL[[m]][j,i]+1
616             isellL=indSellL[[m]][k,i]+1
617             buyLD[[m]][ind,i]=indBuyL[[m]][j,i]+1
618             sellLD[[m]][ind,i]=indSellL[[m]][k,i]+1
619             ind=ind+1
620             break
621           }
622           if ((k==rtime[m])&(indBuyL[[m]][j,i]!=rtime[m])){
623             ibuyL=indBuyL[[m]][j,i]+1
624             isellL=rtime[m]
625             buyLD[[m]][ind,i]=indBuyL[[m]][j,i]+1
626             sellLD[[m]][ind,i]=rtime[m]
627             ind=ind+1
628           }
629         }
630       }}}
631   buyLD[[m]][is.na(buyLD[[m]])]=0
632   buyLD[[m]]=as.matrix(buyLD[[m]])
633   sellLD[[m]][is.na(sellLD[[m]])]=0
634   sellLD[[m]]=as.matrix(sellLD[[m]])
635 }
636
637 # Returns for each pair traded
638 weight11=matrix(nrow=nrow(retoos[[1]]),ncol=10)
639 weight12=matrix(nrow=nrow(retoos[[2]]),ncol=10)
640 weight13=matrix(nrow=nrow(retoos[[3]]),ncol=10)
641 weight14=matrix(nrow=nrow(retoos[[4]]),ncol=10)
642 weight15=matrix(nrow=nrow(retoos[[5]]),ncol=10)
643 weight16=matrix(nrow=nrow(retoos[[6]]),ncol=10)
644 weight17=matrix(nrow=nrow(retoos[[7]]),ncol=10)
645 weight18=matrix(nrow=nrow(retoos[[8]]),ncol=10)
646 weight19=matrix(nrow=nrow(retoos[[9]]),ncol=10)
647 weight1=lapply(paste("weight1", 1:9, sep=''), get)
648
649 weight21=matrix(nrow=nrow(retoos[[1]]),ncol=10)
650 weight22=matrix(nrow=nrow(retoos[[2]]),ncol=10)
651 weight23=matrix(nrow=nrow(retoos[[3]]),ncol=10)
652 weight24=matrix(nrow=nrow(retoos[[4]]),ncol=10)
653 weight25=matrix(nrow=nrow(retoos[[5]]),ncol=10)
654 weight26=matrix(nrow=nrow(retoos[[6]]),ncol=10)
655 weight27=matrix(nrow=nrow(retoos[[7]]),ncol=10)
656 weight28=matrix(nrow=nrow(retoos[[8]]),ncol=10)
657 weight29=matrix(nrow=nrow(retoos[[9]]),ncol=10)
658 weight2=lapply(paste("weight2", 1:9, sep=''), get)
659
660 for(m in 1:length(POOSa)){ind=1
661   for(i in 1:ncol(buyLD[[m]])){
662    for(j in 1:nrow(buyLD[[m]])){
663    if((buyLD[[m]][j,i]!=0)&(buyLD[[m]][j,i]!=sellLD[[m]][j,i])&(buyLD[[m]][j,i]+1!=sellLD[[m]][j,i])&(
         buyLD[[m]][j,i]+2<sellLD[[m]][j,i])){
```

```
664       weight1[[m]][1:((sellLD[[m]][j,i])-(buyLD[[m]][j,i]+1)),ind]=cumprod(retoosw[[m]][(buyLD[[m]][j,i
              ]+2):(sellLD[[m]][j,i]),stock1[1,i,m]])
665       weight2[[m]][1:((sellLD[[m]][j,i])-(buyLD[[m]][j,i]+1)),ind]=cumprod(retoosw[[m]][(buyLD[[m]][j,i
              ]+2):(sellLD[[m]][j,i]),stock2[1,i,m]])
666       ind=ind+1}else{
667     if((buyLD[[m]][j,i]!=0)&(buyLD[[m]][j,i]==sellLD[[m]][j,i])){
668       weight1[[m]][1,ind]=0
669       weight2[[m]][1,ind]=0
670       ind=ind+1}else{
671     if((buyLD[[m]][j,i]!=0)&(buyLD[[m]][j,i]+1==sellLD[[m]][j,i])){
672       weightS1[[m]][1,ind]=0
673       weightS2[[m]][1,ind]=0
674       ind=ind+1} }}
675      }}
676         unity=rep(1, length=ncol(weight1[[m]]))
677         weight1[[m]]=rbind(unity, weight1[[m]])
678         weight2[[m]]=rbind(unity, weight2[[m]])}
679
680 retL1=matrix(nrow=nrow(retoos[[1]]),ncol=4)
681 retL2=matrix(nrow=nrow(retoos[[2]]),ncol=4)
682 retL3=matrix(nrow=nrow(retoos[[3]]),ncol=4)
683 retL4=matrix(nrow=nrow(retoos[[4]]),ncol=4)
684 retL5=matrix(nrow=nrow(retoos[[5]]),ncol=4)
685 retL6=matrix(nrow=nrow(retoos[[6]]),ncol=4)
686 retL7=matrix(nrow=nrow(retoos[[7]]),ncol=4)
687 retL8=matrix(nrow=nrow(retoos[[8]]),ncol=4)
688 retL9=matrix(nrow=nrow(retoos[[9]]),ncol=4)
689 retL=lapply(paste("retL", 1:9, sep=''), get)
690
691 for(m in 1:length(buyLD)){ind=1
692                           for(i in 1:ncol(buyLD[[m]])){
693                             for(k in 1:nrow(buyLD[[m]])){
694                               if(buyLD[[m]][k,i]!=0){
695                                 for(j in 1:((sellLD[[m]][k,i])-(buyLD[[m]][k,i]))){
696                                   retL[[m]][j,ind]=weight1[[m]][j,ind]*retoos[[m]][j+buyLD[[m]][k,i],
                                          stock1[1,i,m]]-weight2[[m]][j,ind]*retoos[[m]][j+buyLD[[m]][k,i
                                          ],stock2[1,i,m]]
697                                 }}else{break}
698                               ind=ind+1
699                             }}
700                           retL[[m]][is.na(retL[[m]])]=0}
701
702 profitLD=matrix(ncol=max(sapply(buyLD,nnzero)), nrow=length(OOSa))
703 for(m in 1:length(OOSa)){
704   for(i in 1:ncol(retL[[m]])){
705     profitLD[m,i]=sum(retL[[m]][,i])
706   }}
707
708 # Information ratio, long
709 IRLD=matrix(nrow=length(OOSa), ncol=max(sapply(buyLD,nnzero)))
710 for(m in 1:length(OOSa)){
711   ind=1
712   for(n in 1:ncol(buyLD[[m]])){
713     for(i in 1:nrow(buyLD[[m]])){
714       if(buyLD[[m]][i,n]!=0){
715         IRLD[m,ind]=mean(retL[[m]][1:(sellLD[[m]][i,n]-buyLD[[m]][i,n]),ind]-PXOOS[[m]][(buyLD[[m]][i,
                n]+1):sellLD[[m]][i,n],3])/sd(retL[[m]][1:(sellLD[[m]][i,n]-buyLD[[m]][i,n]),ind]-PXOOS[[
                m]][(buyLD[[m]][i,n]+1):sellLD[[m]][i,n],3])
716         ind=ind+1
717       }else{break}
718     }}}
719
720 ## Short ##
721 sellSD=NULL
722 buySD=NULL
723 for(m in 1:length(dif)){
724   sellSD[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
```

```
  buySD[[m]]=matrix(nrow=rtime[m],ncol=iakcie[m])
}

for(m in 1:length(dif)){
  for(i in 1:iakcie[m]){
    ind=1
    ibuyS=0
    for(j in 1:rtime[m]){
      if((indSellS[[m]][j,i]>ibuyS)&(indSellS[[m]][j,i]!=rtime[m])){
        for(k in 1:rtime[m]){
          if(indBuyS[[m]][k,i]>=indSellS[[m]][j,i]){
            isellS=indSellS[[m]][j,i]+1
            ibuyS=indBuyS[[m]][k,i]+1
            sellSD[[m]][ind,i]=indSellS[[m]][j,i]+1
            buySD[[m]][ind,i]=indBuyS[[m]][k,i]+1
            ind=ind+1
            break
          }
          if((k==rtime[m])&(indSellS[[m]][j,i]!=rtime[m])){
            isellS=indSellS[[m]][j,i]+1
            ibuyS=rtime[m]
            sellSD[[m]][ind,i]=indSellS[[m]][j,i]+1
            buySD[[m]][ind,i]=ibuyS
            ind=ind+1
          }
        }
      }}}
  sellSD[[m]][is.na(sellSD[[m]])]=0
  sellSD[[m]]=as.matrix(sellSD[[m]])
  buySD[[m]][is.na(buySD[[m]])]=0
  buySD[[m]]=as.matrix(buySD[[m]])
}

# Returns for each pair traded
weightS11=matrix(nrow=nrow(retoos[[1]]),ncol=10)
weightS12=matrix(nrow=nrow(retoos[[2]]),ncol=10)
weightS13=matrix(nrow=nrow(retoos[[3]]),ncol=10)
weightS14=matrix(nrow=nrow(retoos[[4]]),ncol=10)
weightS15=matrix(nrow=nrow(retoos[[5]]),ncol=10)
weightS16=matrix(nrow=nrow(retoos[[6]]),ncol=10)
weightS17=matrix(nrow=nrow(retoos[[7]]),ncol=10)
weightS18=matrix(nrow=nrow(retoos[[8]]),ncol=10)
weightS19=matrix(nrow=nrow(retoos[[9]]),ncol=10)
weightS1=lapply(paste("weightS1", 1:9, sep=''), get)

weightS21=matrix(nrow=nrow(retoos[[1]]),ncol=10)
weightS22=matrix(nrow=nrow(retoos[[2]]),ncol=10)
weightS23=matrix(nrow=nrow(retoos[[3]]),ncol=10)
weightS24=matrix(nrow=nrow(retoos[[4]]),ncol=10)
weightS25=matrix(nrow=nrow(retoos[[5]]),ncol=10)
weightS26=matrix(nrow=nrow(retoos[[6]]),ncol=10)
weightS27=matrix(nrow=nrow(retoos[[7]]),ncol=10)
weightS28=matrix(nrow=nrow(retoos[[8]]),ncol=10)
weightS29=matrix(nrow=nrow(retoos[[9]]),ncol=10)
weightS2=lapply(paste("weightS2", 1:9, sep=''), get)

for(m in 1:length(POOSa)){ind=1
 for(i in 1:ncol(sellSD[[m]])){
  for(j in 1:nrow(sellSD[[m]])){
  if((sellSD[[m]][j,i]!=0)&(sellSD[[m]][j,i]!=buySD[[m]][j,i])&(sellSD[[m]][j,i]+1!=buySD[[m]][j,i])&(
       sellSD[[m]][j,i]+2<buySD[[m]][j,i])){
     weightS1[[m]][1:((buySD[[m]][j,i])-(sellSD[[m]][j,i]+1)),ind]=cumprod(retoosw[[m]][(sellSD[[m]][j
         ,i]+2):(buySD[[m]][j,i]),stock1[1,i,m]])
     weightS2[[m]][1:((buySD[[m]][j,i])-(sellSD[[m]][j,i]+1)),ind]=cumprod(retoosw[[m]][(sellSD[[m]][j
         ,i]+2):(buySD[[m]][j,i]),stock2[1,i,m]])
   ind=ind+1}else{
   if((sellSD[[m]][j,i]!=0)&(sellSD[[m]][j,i]==buySD[[m]][j,i])){
```

```
      weightS1[[m]][1,ind]=0
      weightS2[[m]][1,ind]=0
    ind=ind+1}else{
    if((sellSD[[m]][j,i]!=0)&(sellSD[[m]][j,i]+1==buySD[[m]][j,i])){
      weightS1[[m]][1,ind]=0
      weightS2[[m]][1,ind]=0
  ind=ind+1} }}
  }}
      unity=rep(1, length=ncol(weightS1[[m]]))
      weightS1[[m]]=rbind(unity, weightS1[[m]])
      weightS2[[m]]=rbind(unity, weightS2[[m]])}

retS1=matrix(nrow=nrow(retoos[[1]]),ncol=max(sapply(sellSD, nnzero)))
retS2=matrix(nrow=nrow(retoos[[2]]),ncol=ncol(sellSD[[2]]))
retS3=matrix(nrow=nrow(retoos[[3]]),ncol=ncol(sellSD[[3]]))
retS4=matrix(nrow=nrow(retoos[[4]]),ncol=ncol(sellSD[[4]]))
retS5=matrix(nrow=nrow(retoos[[5]]),ncol=ncol(sellSD[[5]]))
retS6=matrix(nrow=nrow(retoos[[6]]),ncol=ncol(sellSD[[6]]))
retS7=matrix(nrow=nrow(retoos[[7]]),ncol=ncol(sellSD[[7]]))
retS8=matrix(nrow=nrow(retoos[[8]]),ncol=ncol(sellSD[[8]]))
retS9=matrix(nrow=nrow(retoos[[9]]),ncol=ncol(sellSD[[9]]))
retS=lapply(paste("retS", 1:9, sep=''), get)

for(m in 1:length(sellSD)){
  ind=1
  for(i in 1:ncol(sellSD[[m]])){
    for(k in 1:nrow(sellSD[[m]])){
      if(sellSD[[m]][k,i]!=0){
        for(j in 1:((buySD[[m]][k,i])-(sellSD[[m]][k,i]))){
          retS[[m]][j,ind]=-(weightS1[[m]][j,ind]*retoos[[m]][j+sellSD[[m]][k,i],stock1[1,i,m]]-
              weightS2[[m]][j,ind]*retoos[[m]][j+sellSD[[m]][k,i],stock2[1,i,m]])
        }}else{break}
      ind=ind+1
    }}
  retS[[m]][is.na(retS[[m]])]=0}

profitSD=matrix(ncol=max(sapply(sellSD,nnzero)), nrow=length(OOSa))
for(m in 1:length(OOSa)){
  for(i in 1:ncol(retS[[m]])){
    profitSD[m,i]=sum(retS[[m]][,i])
  }}

# Information ratio, short
IRSD=matrix(nrow=length(OOSa), ncol=max(sapply(sellSD,nnzero)))
for(m in 1:length(OOSa)){
  ind=1
  for(n in 1:ncol(sellSD[[m]])){
    for(i in 1:nrow(sellSD[[m]])){
      if(sellSD[[m]][i,n]!=0){
        IRSD[m,ind]=mean(retS[[m]][1:(buySD[[m]][i,n]-sellSD[[m]][i,n]),ind]-PXOOS[[m]][(sellSD[[m]][i
            ,n]+1):buySD[[m]][i,n],3])/sd(retS[[m]][1:(buySD[[m]][i,n]-sellSD[[m]][i,n]),ind]-PXOOS[[
            m]][(sellSD[[m]][i,n]+1):buySD[[m]][i,n],3])
        ind=ind+1
      }else{break}
    }}}
```

D:/RStudio/1daylag.R