１．データセット

```
## 全ﾃﾞｰﾀalldata（1991/1-2004/4の株価，金利，為替のtimeseriesデータ） ##
alldata.r <- getReturns(alldata,type="discrete")  #収益率算出
assetnames <- colnames(alldata.r)

## 事前評価用ﾃﾞｰﾀantedata（1991/1-2003/4の株価，金利，為替のtimeseriesデータ） ##
antedata.r <- getReturns(antedata,type="discrete")

## 事後評価用ﾃﾞｰﾀpostdata 2003/4-2004/4の株価，金利，為替のtimeseriesデータ） ##
postdata <- as.matrix(seriesData(postdata))
temp <- t(postdata[,1:country])/postdata[1,1:country] #株価の初期値を1にする
postdata[,1:country] <- t(temp)
etemp <- postdata[,(country+1):(2*country-1)]
rftemp <- postdata[,(nv-country+2):nv]
rdtemp <- postdata[,"JPY"]
ftemp <- etemp*(1+rdtemp)/(1+rftemp)
postdata.list <- list(Pd=postdata[,1],Pf=postdata[,2:country],e=etemp,rd=rdtemp,f=ftemp)
```

２．事前評価

<u>シナリオ生成</u>

```
NPATH <- 2000                              #ﾊﾟｽ数
NT <- 3                                    #期間数
## 分散共分散行列，平均収益率，初期値 ##
d0 <- postdata[1,]                         #初期値
d0 <- as.matrix(d0)
Q <- var(antedata.r)                       #分散共分散行列
rbar <- apply(antedata.r,2,mean)           #平均収益率
rvar <- apply(antedata.r,2,var)            #分散
## ﾊﾟｽ生成 ##
seed.saved <- .Random.seed
pass.list <- GBM.pass(d0,rbar,rvar,Q)      #幾何ﾌﾞﾗｳﾝ運動によるﾊﾟｽ生成関数
## 最適化へのinput用に変更する ##
senario.list <- senario.data(pass.list) #最適化へのinputﾃﾞｰﾀに変更する関数
## ｱﾒﾘｶﾝｵﾌﾟｼｮﾝ価格，ﾍﾟｲｵﾌ ##
senario.ao <- senario.aoption(pass.list) #最適化へのｱﾒﾘｶﾝｵﾌﾟｼｮﾝ･ﾃﾞｰﾀの生成関数
```

効率的フロンティア

```
nmu <- 9       #繰り返し回数
mu <- 0.025   #要求期待収益率muの初期値
seq <- 0.001  #muの刻み幅
## ﾊﾟﾗﾒｰﾀの設定 ##
alpha <- 0.95     #VaRの水準
W0 <- 10000       #初期国内資金
#mu <- 0.02       #要求期待収益率
tcd <- 0.00005   #取引ｺｽﾄ（国内資産）
tcf <- 0.00005   #取引ｺｽﾄ（外国資産）

# 初期化 #
r.cvar.mat.a <- matrix(NULL,nrow=nmu,ncol=3)              #VaR,CVaR,期待収益率
colnames(r.cvar.mat.a) <- c("VaR","CVaR","期待収益率")
a.ratio.array.a <- array(NULL,dim=c(NT,country+1,nmu))   #投資比率array
dimnames(a.ratio.array.a) <- list(0:(NT-1),c("現金","日本","米国","英国","スイス","カナダ","オー
    ストラリア"),NULL)
h.ratio.array.a <- array(NULL,dim=c(NT,country-1,nmu))   #ヘッジ比率array
dimnames(h.ratio.array.a) <- list(0:(NT-1),c("米国","英国","スイス","カナダ","オーストラリア
    "),NULL)

for(i in 1:nmu){
    module(nuopt,unload=T)  #ヒープメモリをクリアする
    module(nuopt)
    sys.aoption <-
    System(model=multistage.op.aoption,alpha,W0,mu,senario.list,NT,NPATH,tcd,tcf,senario.ao)
    sol.aoption <- solve(sys.aoption)

    # 最適化の結果の取り出し #
    vd0 <- as.array(current(sys.aoption,vd0))
    vd0 <- as.vector(vd0)
    zd0 <- as.array(current(sys.aoption,zd0))
    zd0 <- as.vector(zd0)
    zf0 <- as.array(current(sys.aoption,zf0))
    zf0 <- as.vector(zf0)
    zd <- as.array(current(sys.aoption,zd))
    zd <- as.vector(zd)
    zf <- as.array(current(sys.aoption,zf))
```

```
    vd <- as.array(current(sys.aoption,vd))
    nao <- as.array(current(sys.aoption,nao))
    W <- as.array(current(sys.aoption,W))       #資産
    W1 <- as.array(current(sys.aoption,W1))
    VaR <- as.array(current(sys.aoption,VaR))
    CVaR <- as.array(current(sys.aoption,CVaR))

    r.cvar.mat.a[i,] <- c(VaR,CVaR,mu)
    a.ratio.array.a[,,i] <- a.ratio.result(vd0,zd0,zf0,zd,zf,vd,W,W1,senario.list)
    zf1<-rbind(zf0,zf[-NT,])
    h.ratio.array.a[,,i] <- t(nao)/zf1
    mu <- mu + seq      #muの更新
}
```

３．事後評価

```
## ﾊﾟﾗﾒｰﾀの設定 ##
mu <- 0.025          #要求期待収益率
it <- 12          #運用期間 ヶ月
# ﾛｰﾘﾝｸﾞ統計量 #
rolingmean <- aggregateSeries(alldata.r,moving=nrow(alldata.r)-it,adj=1,FUN=mean)
rolingmean <- as.matrix(seriesData(rolingmean))
rolingvar <- aggregateSeries(alldata.r,moving=nrow(alldata.r)-it,adj=1,FUN=var)
rolingvar <- as.matrix(seriesData(rolingvar))
rolingvar.mat <-
    aggregateSeries(alldata.r,together=T,moving=nrow(alldata.r)-it,adj=1,FUN=var)
rolingvar.mat <- as.matrix(seriesData(rolingvar.mat))

# ｱﾒﾘｶﾝｵﾌﾟｼｮﾝの行使ﾚｰﾄ #
a <- 1+rdtemp
a2 <- 1+rftemp
for(i in 1:(ni-2)){
    a[i] <- a[i]*a[i+1]*a[i+2]
    a2[i,] <- a2[i,]*a2[i+1,]*a2[i+2,]
}
a[ni-1] <- a[ni-1]*a[ni]
a2[ni-1,] <- a2[ni-1,]*a2[ni,]
f3 <- etemp*a/a2
K0 <- f3[-ni,]
```

```
## 初期化 ##
V.a <- c(W0,rep(NA,it))                                #資産ﾍﾞｸﾄﾙ初期化
a.ratio.a <- matrix(NA,nrow=it,ncol=country+1) #資産配分比率
colnames(a.ratio.a) <- c("現金",assetnames[1:country])
h.ratio.a <- matrix(NA,nrow=it,ncol=country-1) #ﾍｯｼﾞ比率
colnames(h.ratio.a) <- assetnames[2:country]
nao.mat <- matrix(0,ncol=country-1,nrow=it)     #ｵﾌﾟｼｮﾝ価格行列
ao.cf <- array(0,dim=c(it,it+2,country-1))      #ｵﾌﾟｼｮﾝ･ﾍﾟｲｵﾌ配列

for(i in 1:it){
    ## 初期値，平均収益率，分散，分散共分散行列の設定 ##
    b0 <- postdata[i,]
    b0 <- as.matrix(b0)
    Q1 <- matrix(rolingvar.mat[i,],nrow=nv,ncol=nv)    #分散共分散行列
    rbar1 <- rolingmean[i,]        #平均収益率
    rvar1 <- rolingvar[i,]         #分散
    # ﾊﾟｽ生成 #
    .Random.seed <- seed.saved
    pass <- GBM.pass(b0,rbar1,rvar1,Q1)
    # 最適化へのinputの型に変更する #
    senario.list1 <- senario.data(pass)
    # ｱﾒﾘｶﾝｵﾌﾟｼｮﾝ価格，ﾍﾟｲｵﾌ #
    senario.ao1 <- senario.aoption(pass)
    # 最適化 #
    module(unload=T)
    module(nuopt)
    sys.aoption <-
        System(model=multistage.op.aoption,alpha,W0,mu,senario.list1,NT,NPATH,tcd,tcf,senario.ao1)
    sol.aoption <- solve(sys.aoption)

    # 最適化の結果の取り出し #
    vd0 <- as.array(current(sys.aoption,vd0))
    vd0 <- as.vector(vd0)
    zd0 <- as.array(current(sys.aoption,zd0))
    zd0 <- as.vector(zd0)
    zf0 <- as.array(current(sys.aoption,zf0))
    zf0 <- as.vector(zf0)
    nao <- as.array(current(sys.aoption,nao))
```

```
    nao0 <- nao[,1]
    nao.mat[i,] <- nao0
    Pd0 <- b0[1]
    Pf0 <- b0[2:country]
    e0 <- b0[(country+1):(2*country-1)]
    a.ratio.a[i,]<-c(vd0,Pd0*zd0,Pf0*e0*zf0)/V.a[i]
    h.ratio.a[i,]<-nao0/zf0

    apay <- senario.ao1$payoff.a[,-NT,,1]
    rda<-pass$rd[,c(-1,-(NT+1))]
    rda.temp <- array(rep(rda,country-1),dim=c(NPATH,NT-1,country-1))
    cf <- apay*exp(-rda.temp)
    cfbar <- apply(cf,c(2,3),mean)
    ea <- postdata.list$e[-1,]
    for(m in 1:(country-1)){
        if(K0[i,m]-ea[i,m]>cfbar[1,m]){
            ao.cf[i,i,m] <- K0[i,m]-ea[i,m]
        }
        for(j in 2:(NT-1)){
            if(sum(ao.cf[i,,m]==0)==0){
                if(K0[i,m]-ea[i+j-1,m]>cfbar[j,m]){
                    ao.cf[i,i+j-1,m] <- K0[i,m]-ea[i+j-1,m]
                }
            }
        }
        if(sum(ao.cf[i,,m]==0)==0){
            if(K0[i,m]-ea[i+NT-1,m]>0){
                ao.cf[i,i+NT-1,m] <- K0[i,m]-ea[i+NT-1,m]
            }
        }
    }
    V.a[i+1] <- postdata.list$Pd[i+1]*zd0*(1-tcd) + (1+postdata.list$rd[i])*vd0 +
        sum(postdata.list$Pf[i+1,]*postdata.list$e[i+1,]*zf0)*(1-tcf)+sum(nao.mat*ao.cf[,i,
        ])
    W0 <- V.a[i+1]
}
```

４．関数の定義

最小二乗ﾓﾝﾃｶﾙﾛ法 LSM_put関数定義

```
LSM.put <- function(stock,K,rf,ORDER){
npath <- nrow(stock)                                          #ｼﾐｭﾚｰｼｮﾝ数
NT <- ncol(stock)                                             #期間数（0時点を含む）
option.cashflow <- matrix(NA,nrow=npath,ncol=NT-1)
cashflow <- K-stock[,NT]                                      #満期ｷｬｯｼｭﾌﾛｰ,max(K-S,0)
cashflow[cashflow<0] <- 0

for(t in (NT-1):2){
   exerval <- K-stock[,t]                                     #行使価値,max(K-S,0)
   exerval[exerval<0] <- 0
   x <- stock[,t][exerval>0]                                  #説明変数
   y <- cashflow[exerval>0]*exp(-rf[,t][exerval>0])           #目的関数
   kaiki <- lm(y~poly(x,ORDER))                               #ORDER次の多項式による回帰
   contval <- exerval                                         #継続価値
   contval[contval>0] <- kaiki$fitted.values
   option.cashflow[,t] <- cashflow
   cashflow <- exerval                                        #cashflow更新
   cashflow[exerval<contval] <- 0
}

# 1時点のｵﾌﾟｼｮﾝ行使及びｷｬｯｼｭﾌﾛｰの決定 #
cashflow <- exerval
cashflow[exerval<contval] <- 0
option.cashflow[,1] <- cashflow

# 行使後の価値は0 #
for(i in 2:(NT-1)){
   for(j in 1:npath){
      if(sum(option.cashflow[j,1:(i-1)]>0)>0){
         option.cashflow[j,i] <- 0
      }
   }
}
# ｵﾌﾟｼｮﾝ価格算出，ｵﾌﾟｼｮﾝｷｬｯｼｭﾌﾛｰの割引 #
temp <- option.cashflow[,NT-1]*exp(-rf[,NT-1])+option.cashflow[,NT-2]
if(NT>3){
```

```
    for(t in (NT-2):2){
        temp <- temp*exp(-rf[,t])+option.cashflow[,t-1]
    }
}
temp <- temp*exp(-rf[,1])
option.price <- mean(temp)
return(option.price,option.cashflow)
}
```

モンテカルロ法によるヨーロピアンオプションの評価関数

```
M.e.put <- function(stock,K,rf){
    NT <- ncol(stock)
    if(ncol(rf)==2){
        rf <- sum(rf[,-NT])
    }else{
        rf <- apply(rf[,-NT],1,sum)
    }
    option.cashflow <- K-stock[,NT]
    option.cashflow[option.cashflow<0] <- 0
    temp <- option.cashflow*exp(-rf)
    option.price <- mean(temp)
    return(option.price,option.cashflow)
}
```

パス生成（幾何ブラウン運動）の関数定義

```
GBM.pass <- function(d0,rbar,rvar,Q){
    epsilon <- rmvnorm(NPATH*NT,cov=Q)                  #多変量正規分布
    epsilon.array <- array(epsilon,dim=c(NPATH,NT,nv))
    pass.array <- array(NA,dim=c(NPATH,NT+1,nv))        #パス配列初期化
    for(i in 1:NT){
        for(j in 1:nv){
            pass.array[,1,j] <- d0[j]
            pass.array[,i+1,j] <-
    pass.array[,i,j]*exp(rbar[j]-0.05*rvar[j]+epsilon.array[,i,j])
        }
    }
    Pd <- pass.array[,,1]
    Pf <- pass.array[,,2:country]
```

```
    rd <- pass.array[,,nv-country+1]
    rf <- pass.array[,,(nv-country+2):nv]
    rd.temp <- array(rep(rd,country-1),dim=c(NPATH,NT+1,country-1))
    e <- pass.array[,,(country+1):(nv-country)]
    f <- e*(1+rd.temp)/(1+rf)
    pass.list <- list(Pd=Pd,Pf=Pf,rd=rd,rf=rf,e=e,f=f)
    pass.list
}
```

最適化へのinputの型に変更する関数

```
senario.data <- function(pass.list){
    Pd0 <- pass.list$Pd[1,1]
    Pf0 <- pass.list$Pf[1,1,]
    e0 <- pass.list$e[1,1,]
    rd0 <- pass.list$rd[1,1]
    rf0 <- pass.list$rf[1,1,]
    f0 <- pass.list$f[1,1,]
    Pd <- pass.list$Pd[,-1]        #初期値を除く
    Pf <- pass.list$Pf[,-1,]
    e <- pass.list$e[,-1,]
    f <- pass.list$f[,-1,]
    rd <- pass.list$rd[,-1]
    fvar <- apply(f,c(2,3),mean)
    list(rd0=rd0,Pd0=Pd0,Pf0=Pf0,e0=e0,f0=f0,rd=rd,Pd=Pd,Pf=Pf,e=e,f=f,fvar=fvar)
}
```

最適化へのｱﾒﾘｶﾝ･ｵﾌﾟｼｮﾝﾃﾞｰﾀ算出関数（LSMの適用）

```
senario.aoption <- function(pass.list){
# 権利行使価格(為替ﾌｫﾜｰﾄﾞﾚｰﾄ) #
Ka.mat <- matrix(NA,nrow=country-1,ncol=NT) #権利行使価格行列
    # 満期の長い為替ﾌｫﾜｰﾄﾞﾚｰﾄ算出 #
        # 金利ﾌｫﾜｰﾄﾞﾚｰﾄ算出 #
rd.temp <- 1+pass.list$rd[,1:NT]
rf.temp <- 1+pass.list$rf[,1:NT,]
for(i in (NT-1):1){
    rd.temp[,i] <- rd.temp[,i]*rd.temp[,i+1]
    rf.temp[,i,] <- rf.temp[,i,]*rf.temp[,i+1,]
}
```

```
rd <- rd.temp[,1:(NT-1)]       #国内金利ﾌｫﾜｰﾄﾞﾚｰﾄ行列
rf <- rf.temp[,1:(NT-1),]      #外国金利ﾌｫﾜｰﾄﾞﾚｰﾄ行列
temp <- array(rep(rd,country-1),dim=c(NPATH,NT-1,country-1))
e <- pass.list$e[,1:(NT-1),]
f <- e*temp/rf                 #為替ﾌｫﾜｰﾄﾞﾚｰﾄ行列
    # 権利行使価格 #
Ka.mat[,NT] <- apply(pass.list$f[,NT,],2,mean)
Ka.mat[,1:(NT-1)] <- t(apply(f,c(2,3),mean))

# 原資産ﾘｽﾄ, 無ﾘｽｸﾚｰﾄﾘｽﾄ #
S.list <- list(pass.list$e)
rd.list <- list(pass.list$rd)
for(i in 2:NT){
    temp <- pass.list$e[,i:(NT+1),]
    temp2 <- pass.list$rd[,i:(NT+1)]
    S.list <- c(S.list,list(temp))
    rd.list <- c(rd.list,list(temp2))
}

# ｱﾒﾘｶﾝ･ｵﾌﾟｼｮﾝ価格, ﾍﾟｲｵﾌ #
Pao <- matrix(NA,ncol=NT,nrow=country-1)           #ｵﾌﾟｼｮﾝ価格行列
payoff.a <- array(0,dim=c(NPATH,NT,country-1,NT)) #ｵﾌﾟｼｮﾝのﾍﾟｲｵﾌ配列
    # LSMの適用    #
for(m in 1:(country-1)){
    for(i in 1:(NT-1)){
        option.temp <- LSM.put(S.list[[i]][,,m],Ka.mat[m,i],rd.list[[i]],2)
        Pao[m,i] <- option.temp$option.price
        payoff.a[,i:NT,m,i] <- option.temp$option.cashflow
    }
}
    # 最終時点-1時点はﾖｰﾛﾋﾟｱﾝﾌﾟｯﾄ価格 #
for(m in 1:(country-1)){
    option.temp2 <- M.e.put(S.list[[NT]][,,m],Ka.mat[m,NT],rd.list[[NT]])
    Pao[m,NT] <- option.temp2$option.price
    payoff.a[,NT,m,NT] <- option.temp2$option.cashflow
}
# ｵﾌﾟｼｮﾝ番号を0から始める #
dimnames(payoff.a) <- list(NULL,NULL,NULL,0:(NT-1))
```

```
colnames(Pao) <- 0:(NT-1)
list(Pao=Pao,payoff.a=payoff.a)
}
```

アメリカンオプション・ヘッジモデル（シミュレーション型多期間最適化モデル）

```
multistage.op.aoption = function(alpha,W0,mu,senario,NT,NPATH,tcd,tcf,senario.ao){

#集合の設定
f.Market.set <- Set()
Time.set <- Set()
Senario.set <- Set()
Option.set <- Set()

#要素の設定
m <- Element(set =f.Market.set)
t <- Element(set = Time.set)
s <- Element(set = Senario.set)
j <- Element(set = Option.set)

#パラメータの設定
alpha <- Parameter(alpha)            #VaRの水準
W0 <- Parameter(W0) #初期富
mu <- Parameter(mu)
Pd0 <- Parameter(senario$Pd0)              #資産の初期価格
Pf0 <- Parameter(as.array(senario$Pf0),index=m)
Pd <- Parameter(senario$Pd,index=dprod(s,t))   #資産の価格
Pf <- Parameter(senario$Pf,index=dprod(s,t,m))
e0 <- Parameter(as.array(senario$e0),index=m)          #初期為替レート
e <- Parameter(senario$e,index=dprod(s,t,m))   #為替レート
rd0 <- Parameter(senario$rd0) #初期国内金利
rd <- Parameter(senario$rd,index=dprod(s,t))   #国内金利
tcd <- Parameter(tcd)   #取引コスト
tcf <- Parameter(tcf)
#Pao0 <- Paremeter(Pao0,index=m)
Pao <- Parameter(senario.ao$Pao,index=dprod(m,j)) #オプション価格
payoff <- Parameter(senario.ao$payoff.a,index=dprod(s,t,m,j))   #オプションのペイオフ

#決定変数
```

```
vd0 <- Variable()    #国内通貨運用量
zd0 <- Variable()    #国内株式運用量
zf0 <- Variable(index=m)    #外国株式運用量

vd <- Variable(index=dprod(s,t))
zd <- Variable(index=t)
zd.b <- Variable(index=t)  #国内株式購入量
zd.s <- Variable(index=t)  #国内株式売却量
zf <- Variable(index=dprod(t,m))
zf.b <- Variable(index=dprod(t,m))    #外国株式購入量
zf.s <- Variable(index=dprod(t,m))    #外国株式売却量
nao <- Variable(index=dprod(m,j)) #ｵﾌﾟｼｮﾝ購入量

VaR <- Variable()
y <- Variable(index=s)

#Expressionの設定
W1 <- Expression(index=s)
W1[s] ~ Pd[s,1]*zd0*(1-tcd) + (1+rd0)*vd0 + Sum(Pf[s,1,m]*zf0[m]*e[s,1,m]*(1-tcf) +
    nao[m,0]*payoff[s,1,m,0],m)
W <- Expression(index=dprod(s,t))
W[s,t,t>=2] ~ Pd[s,t]*zd[t-1]*(1-tcd) + (1+rd[s,t-1])*vd[s,t-1] +
    Sum(Pf[s,t,m]*zf[t-1,m]*e[s,t,m]*(1-tcf) + Sum(nao[m,j]*payoff[s,t,m,j],j,j<=t-1),m)

#目的関数
CVaR <- Objective(type="minimize")
CVaR ~ VaR + Sum(y[s],s)/(NPATH*(1-alpha))

    #制約式の設定
#0時点での配分決定
Pd0*zd0*(1+tcd) + vd0 + Sum(Pf0[m]*zf0[m]*e0[m]*(1+tcf) + nao[m,0]*Pao[m,0],m) == W0

#投資量保存式
zd[1] == zd0 + zd.b[1] -zd.s[1]
zf[1,m] == zf0[m] + zf.b[1,m] -zf.s[1,m]
zd[t,t>=2,t<=NT-1] == zd[t-1] + zd.b[t] - zd.s[t]
zf[t,m,t>=2,t<=NT-1] == zf[t-1,m] + zf.b[t,m] - zf.s[t,m]
```

#ｷｬｯｼｭ･ﾊﾞﾗﾝｽ条件

```
zd.b[1]*Pd[s,1]*(1+tcd) + Sum(zf.b[1,m]*Pf[s,1,m]*e[s,1,m]*(1+tcf) + nao[m,1]*Pao[m,1],m) +
    vd[s,1] == zd.s[1]*Pd[s,1]*(1-tcd) + (1+rd0)*vd0 +
    Sum(zf.s[1,m]*Pf[s,1,m]*e[s,1,m]*(1-tcf) + nao[m,0]*payoff[s,1,m,0],m)
zd.b[t,t>=2,t<=NT-1]*Pd[s,t]*(1+tcd) + Sum(zf.b[t,m]*Pf[s,t,m]*e[s,t,m]*(1+tcf) +
    nao[m,t]*Pao[m,t],m) + vd[s,t] == zd.s[t]*Pd[s,t]*(1-tcd) + (1+rd[s,t-1])*vd[s,t-1] +
    Sum(zf.s[t,m]*Pf[s,t,m]*e[s,t,m]*(1-tcf) + Sum(nao[m,j]*payoff[s,t,m,j],j,j<=t-1),m)

(Sum(W[s,NT],s)/NPATH)/W0-1 >= mu

vd0 >= 0
zd0 >= 0
zf0[m] >= nao[m,0] >= 0
vd[s,t,t<=NT-1] >= 0
zd.b[t,t<=NT-1] >= 0
zf.b[t,m,t<=NT-1] >= 0
zd.s[1] >= 0
zf.s[1,m] >= 0
zd[t-1] >= zd.s[t,t>=2,t<=NT-1] >= 0
zf[t-1,m] >= zf.s[t,m,t>=2,t<=NT-1] >= 0
zd[NT-1] >= 0
(Sum(Pf[s,t,m],s)/NPATH)*zf[t,m,t<=NT-1] >= nao[m,t] >= 0

y[s] + VaR + (W[s,NT]/W0-1) >= 0
y[s] >= 0
}
```

資産投資比率算出関数

```
a.ratio.result <- function(vd0,zd0,zf0,zd,zf,vd,W,W1,senario){
    vdbar <- apply(vd[,-NT],2,mean)   #現金
    vvd <- c(vd0,vdbar)
    vzd <- c(zd0,zd[-NT])   #投資量
    pd1 <- senario$Pd[,-NT]
    pd1bar <- apply(pd1,2,mean)
    vpd1 <- c(senario$Pd0,pd1bar) #平均価格
    vpd <- vpd1*vzd  #平均投資額

    zf1 <- zf[-NT,]
```

```
    vzf <- rbind(zf0,zf1)   #投資量
    pf1 <- senario$Pf[,-NT,]
    e1 <- senario$e[,-NT,]
    pfe1 <- pf1*e1
    pfe1bar <- apply(pfe1,c(2,3),mean)
    vpfe1 <- rbind(senario$e0*senario$Pf0,pfe1bar)
    vpf <- vpfe1*vzf

    mat.a <- rbind(vvd,vpd,t(vpf))

    # 平均投資比率算出 #
    W[,1] <- W1
    w1 <- W[,-NT]
    wbar <- apply(w1,2,mean)
    vw1 <- c(W0,wbar)    #平均富
    mat.r <- t(mat.a)/vw1   #投資比率
    mat.r
}
```