```
# MGARCH.ssc
# Author: Jiahui (Jeff) Wang
# Prepared for the MGARCH Chapter
# Date: April 5, 2002
# Updated: July 6, 2007 by Eric Zivot

# EWMA covariance estimations

hp.ibm=seriesMerge(hp.s,ibm.s)
tmp=acf(hp.ibm^2)

hp.ibm.cov=EWMA.cov(hp.ibm,lambda=0.9672375)
seriesPlot(cbind(hp.ibm.cov[,1,1],hp.ibm.cov[,2,2],hp.ibm.cov[,1,2]),
one.plot=F,strip.text=c("HP Vol.","IBM Vol.","Cov."))

hp.ibm.ewma=mgarch(hp.ibm~1,~ewma1,trace=F)
hp.ibm.ewma

mgarch(hp.ibm~1,~ewma2,trace=F)

# DVEC model and diagnostics

hp.ibm.dvec=mgarch(hp.ibm~1,~dvec(1,1),trace=F)
class(hp.ibm.dvec)
hp.ibm.dvec

names(hp.ibm.dvec)
coef(hp.ibm.dvec)

sqrt(diag(vcov(hp.ibm.dvec,method="qmle")))
residuals(hp.ibm.dvec, standardize=T)

summary(hp.ibm.dvec)

autocorTest(residuals(hp.ibm.dvec,standardize=T)^2,lag=12)
archTest(residuals(hp.ibm.dvec,standardize=T),lag=12)
autocorTest(residuals(hp.ibm.dvec,standardize=T)^2,lag=12,bycol=F)

plot(hp.ibm.dvec, ask=F)

# plot conditional correlations
hp.ibm.cross=hp.ibm.dvec$R.t[,1,2]
hp.ibm.cross=timeSeries(hp.ibm.cross,pos=positions(hp.ibm))
seriesPlot(hp.ibm.cross,strip="Conditional Cross Corr.")

# Matrix diagonal model
mgarch(hp.ibm~1,~dvec.mat.scalar(1,1),trace=F)

# BEKK model
hp.ibm.bekk=mgarch(hp.ibm~1,~bekk(1,1))
hp.ibm.bekk

# compare DVEC with BEKK
seriesPlot(cbind(hp.ibm.dvec$R.t[,1,2],hp.ibm.bekk$R.t[,1,2]),
           strip=c("DVEC Corr.","BEKK Corr."),one.plot=F,layout=c(1,2,1))

# univariate GARCH-based models

# CCC with 2-component univariate garch
mgarch(hp.ibm~1,~ccc.two.comp(1,1),trace=F)

# principle component garch with 2-component univariate garch
mgarch(hp.ibm~1,~prcomp.pgarch(1,1,1),trace=F)

# pure diagonal model with univariate egarch
mgarch(hp.ibm~1,~egarch(1,1),leverage=T,trace=F)

# MGARCH with exogenous variables
hp.ibm.beta=mgarch(hp.ibm~seriesData(nyse.s),~dvec(1,1),xlag=1)
summary(hp.ibm.beta)

plot(hp.ibm.beta, ask=F)

weekdaysVec=as.integer(weekdays(positions(hp.ibm)))
MonFriDummy=(weekdaysVec==2|weekdaysVec==6)
hp.ibm.dummy=mgarch(hp.ibm~1,~dvec(1,1)+MonFriDummy)
summary(hp.ibm.dummy)

# multivariate garch with non-normal errors
hp.ibm.dvec.t=mgarch(hp.ibm~1,~dvec(1,1),cond.dist="t")
hp.ibm.dvec.t$cond.dist
```

```
# compare models
hp.ibm.comp=compare.mgarch(hp.ibm.dvec, hp.ibm.dvec.t)
hp.ibm.comp
plot(hp.ibm.comp,qq=T)

# Prediction from MGARCH models
hp.ibm.pred = predict(hp.ibm.bekk, 10)
class(hp.ibm.pred)
names(hp.ibm.pred)
plot(hp.ibm.pred, ask=F, which.plot=2, layout=c(1,2))

# custom estimation of MGARCH models

class(hp.ibm.dvec$model)
hp.ibm.dvec$model
names(hp.ibm.dvec$model)
hp.ibm.dvec$model$arch


bekk.mod=hp.ibm.bekk$model
bekk.mod$a.value[2,1]=0
hp.ibm.bekk2=mgarch(series=hp.ibm,model=bekk.mod)


bekk.mod=hp.ibm.bekk$model
bekk.mod$c.value=rep(0,2)
bekk.mod$c.which=rep(F,2)
hp.ibm.bekk3=mgarch(series=hp.ibm, model=bekk.mod)
LR.stat=-2*(hp.ibm.bekk3$likelihood - hp.ibm.bekk$likelihood)
LR.stat

# multivariate garch model simulation
hp.ibm.sim = simulate(hp.ibm.bekk, n=10)
class(hp.ibm.sim)
names(hp.ibm.sim)
hp.ibm.sim$et
hp.ibm.sim$V.t

# simulation-based forecasts
eps.start = residuals(hp.ibm.bekk)[2000,]@data
V.start = hp.ibm.bekk$S.t[2000, , ]
n.rep = 200
hp.ibm.sim = array(0, c(100, 2, n.rep))
set.seed(10)
for (i in 1:n.rep) {
   eps.pred = rbind(eps.start, rmvnorm(100))
   tmp = simulate(hp.ibm.bekk, n=100, n.start=0,
         etat=eps.pred, V.start=V.start)$V.t
   hp.ibm.sim[, , i] = matrix(tmp,byrow=T,nrow=100)[,c(1,4)]
}
hp.ibm.sim = sqrt(hp.ibm.sim)
hp.ibm.simpred = rowMeans(hp.ibm.sim, dims=2)
hp.ibm.simstde = rowStdevs(hp.ibm.sim, dims=2)
```