I.ABSTRACT

Value at Risk (VaR) plays a central role in risk management. Nowadays, the copula theory is a fundamental tool in modeling multivariate distributions. In this project, our purpose is to estimate VaR with AR-GARCH and Copula model. Firstly, we analyze historical data of Nasdaq and S&P500 from 2004-2014. Secondly, fit the data using GARCH-N and GARCH-t models to generate copula parameters. Thirdly, we use 750 days moving window to backtest the above models and compare the VaR with actual loss. We figure out the best models under various conditions for evaluating the value at risk of portfolio.

II. INTRODUCTION

The financial world is changeable and unpredictable, making the risk management increasingly important. Value at Risk (VaR) is of great importance in risk management. Traditionally, there are following methods for the estimation of VaR, such as the historical simulation, the variance-covariance and the Monte Carlo approaches. The first one does not assume any distribution, the last two approaches demand the joint distribution, which in the analytical approach is frequently the normal distribution.

Copula is a useful method for modeling multivariate distributions. A copula model is the dependence between the variates in a multivariate distribution through the marginal distributions. Consequently, the use of copula allows us to take advantage of the wide variety of univariate models instead of the joint distributions of the variates.

The copula theory has been widely used in estimation in different marginal distribution cases. Time variation is widely discussed in the literature before, so the combination of application of the time series distribution and copula seems natural in finance statistics.

The stock market is an indispensable part of the financial world. We want to apply time series and copula theory to make a research on the stock prices and do the estimation of VaR of a portfolio composed by Nasdaq and S&P500 stock indices.

III.& IV. ANALYSIS AND RESULTS INTERPRETATION

A. Data

We used a portfolio consists of Nasdaq and S&P 500 stock indices and took the log returns of their daily closing prices from Jan 1st, 2004 to Nov 29th, 2014. And they were denoted as variable 1 (X1) and variable 2 (X2). Analyzing plots of their returns and absolute returns (see Attachment), we could conclude the features of volatility clustering, large kurtosis and symmetric distributions.

Table1: Descriptive Statistics

Statistics	**Nasdaq**	**S&P500**
Mean	0.000316966	0.000227014
Standard Deviation	0.01352728	0.012487451
Minimum	-0.095876904	-0.094695145
Median	0.000939093	0.000751025
Maximum	0.111594417	0.109571959
Excess of Kurtois	7.381758433	11.60642557
Skewness	-0.2406668	-0.34229321

B. Time Series Models and Modeling the marginal distributions

The table 2 indicates the appropriateness of using time series to model the two stock indices' returns. Research has shown that return series can be successfully modeled by ARMA-GARCH model. Having the estimated parameters being significant at 5% confidence level, we chose AR(1)-GARCH(1,1) model for the stock indices. The model is presented as following:

$X_{i,\,t} = \mu_i + \varphi_i\, X_{i,\,t-1} + \varepsilon_{i,\,t}$

$\varepsilon_{i,\,t} = \sigma_{i,\,t}\eta_{i,\,t}$

$\sigma^2{}_{i,\,t} = \alpha_i + \beta_i\, \varepsilon^2{}_{i,\,t-1} + \gamma_i\, \sigma^2{}_{i,\,t-1},$ *where* i= 1, 2 and β_i+γ_i < 1……………………………(1.1)

We assume that the conditional innovations $\eta_{i,\,t} = \varepsilon_{i,\,t}\,\sigma_{i,\,t}|F_{\ i,\,t-1},\ i$= 1, 2 follow standard normal distribution and standard t distribution. The corresponding GARCH models are denoted by GARCH-N and GARCH-T. We use the two models as filters to obtain innovation vectors $\{\eta_{1,\,t},\ \eta_{2,\,t}\}$, whose distributions are going to be modeled by copulas later. The following table presents the result of parameters estimation for the GARCH-N and GARCH-T models:

Table 2: Parameter estimates of GARCH models and standard errors

	Parameters	**GARCH-N**	**se1**	**GARCH-t**	**se2**
Nasdaq	**u1**	0.000715	0.000186	0.000949	0.000183
	phi1	-0.0271	0.0202	-0.0266	0.0194
	alpha1	0.00000248	0.000000544	0.00000213	0.000000597
	beta1	0.0834	0.00991	0.0849	0.0119
	gamma1	0.899	0.0114	0.902	0.0129
	v1			8.53	1.43
	AIC	-6.179562		-6.2	
S&P500	**u1**	0.000589	0.000151	0.000816	0.000143
	phi1	-0.0616	0.0206	-0.0593	0.0189

alpha1	0.00000182	0.000000353	0.00000147	0.000000409
beta1	0.0955	0.0106	0.0992	0.0135
gamma1	0.888	0.0116	0.892	0.0134
v1			6.03	0.765
AIC	-6.521793			-6.559263

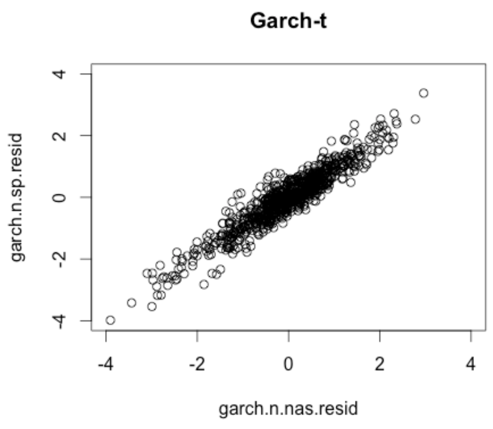
The dependence of returns is captured by the residuals of the two models. In order to further test the goodness of fit for the GARCH-N and GARCH-t models, we conducted the Ljung–box test. To test the null hypothesis that the residuals of AR(1) are independent. We obtained that p-values are way greater than .05 in GARCH-t and GARCH-N cases for residual and residual^2. Hence, we could conclude that the two models fit well.

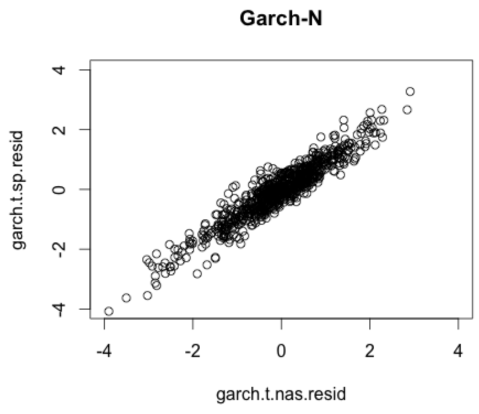
Table 3:Ljung-Box test (α=0.05)

	GARCH-N Nasdaq	GARCH-t Nasdaq	GARCH-N S&P500	GARCH-t S&P500
R	0.9178132	0.9065603	0.05044402	0.08369201
R^2	0.5876733	0.3837712	0.9607865	0.8629479

C. Fitting model for different copulas

First of all, we need to check the dependencies of the two series in GARCH-N and GARCH-t, we can draw the scatterplots as below, it shows that there is a positive dependence between the two series.





Here, since Gumbel and Frank Copula have outstanding performance in finance statistics, we will use the Student-t Copula, Clayton Copula, Gumbel Copula, Frank Copula and Plackett Copula with Student-t and Normal marginal distributions. Here, for the marginal distributions, we use the residuals of the adjustment of the GARCH-N and GARCH-t for the normal and t distributions. The table below shows the result of the copula parameters using maximum likelihood method.

Table4: Parameter estimates and standard errors for Plackett, Student-t, Gumbel, Frank, Clayton copulas

Copulas	Parameter	GARCH-N	GARCH-t
Placket	**theta**	95.49693	95.00351
Studen-t	**v**	14.91775	8.479518
	se	3.429257	1.593718
	R12	0.9397674	0.9511967
	se	0.00178824	0.001553246
Gaumbel	**param**	4.314239	4.821897
	se	0.06843129	0.07717385
Frank	**param**	16.28931	17.07474
	se	0.2918175	0.3038215
Clayton	**param**	3.848723	4.861411
	se	0.08688641	0.1014028

In order to select the copula function, we measure the quadratic distance between the estimated copula and the empirical copula in the region of interest. That is, when selecting the model to estimating the VaR, we need to take the lower tail into consideration. We will use the region of $[0, \gamma]^2$ for $\gamma = 0.05$, 0.1 and 0.2. The table below shows quadratic distance between the estimation copula and the empirical copula.

Table5:Quadratic distance between the estimated and empirical copulas in the left tail (square [0, γ]2)

	Clayton		Gumbel		Frank		Student t		Plackett	
	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T
γ=0.05	0.06354	0.06027	29.90746	29.6756	0.07187	0.07109	0.24476	0.26745	0.54579	0.51067
γ=0.1	0.06465	0.06146	7.46584	7.27378	0.06901	0.0681	0.67986	0.68375	2.27613	2.18934
γ=0.2	0.04162	0.03907	1.75679	1.75658	0.04513	0.0444	3.86827	3.95855	7.78898	7.74834

D. Estimating VaR

In this paper we construct a portfolio with equal weights of S&P500 and NASDAQ indices. The equal weight does not serve as a constraint but can vary freely. The portfolio series has the form that $X_{p,t} = 1/2X_{1,t} + 1/2X_{2,t}$.

We estimate VaR using the backtesting method. The moving window is set to be 800 days, in which the in-sample size is 750 days and the out-sample size is 50 days. The windows move forward every 50 days. With the in sample data, we estimate its AR(1)-GARCH(1,1) model, whose residuals are applied to estimate copula's parameter. After the total models are obtained, we generate innovation vectors for the 50 following days, with 5000 repeated simulations each day. More specifically, innovation vectors are probability vectors that are generated randomly from each copula, and following (1.1) we can get an estimation of the joint distribution of them. Using the joint distribution we can have the distribution of portfolio and VaR at each day in the in-sample set. VaR is evaluated for alpha =0.1 and -.05. We have a total of 2647 time series observations and have 2647 – 750 = 1897 VaRs tested.

The value in the table below are obtained this way: for each level (t), comparing the simulated portfolio return $\hat{X}_{p,t}$ to the real return $X_{p,t}$ and counting the number of those above observed $X_{p,t}$. Proportions are displaced with numbers in the brackets.

Table 6: Proportion of observations (number of observations in brackets), for t=751 to t=2747,where the portfolio loss exceed the estimated VaR for α=0.05 and α=0.1.

	Clayton		Gumbel		Frank		Student t		Plackett	
	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T	GARCH-N	GARCH-T
α=0.05	131 (0.0694)	76 (0.0401)	134 (0.0707)	**93** (**0.0491**)	123 (0.0649)	92 (0.0486)	133 (0.0703)	91 (0.0482)	137 (0.0725)	89 (0.0473)
α=0.1	208 (0.11)	168 (0.089)	218 (0.115)	169 (0.0896)	**194** **(0.1027)**	171 (0.0905)	199 (0.10540)	148 (0.0784)	197 (0.10405)	161 (0.0851)

The results shows that at alpha =0.1, VaR of GARCH-N Frank copula is the closest to its confidence level; which at alpha = 0.05, VaR of GARCH-t Gumbel copula is the closest to its confidence level. It is also observed that all GARCH-N models have VaR that's higher than its corresponding alpha while all GARCH-t models have the opposite outcomes. Generally speaking, at a relative higher level of alpha, GARCH-N Frank copula gives a better estimation of VaR; on the other hand, at a relative lower level of alpha, GARCH-t Gumbel copula gives a better estimation.

V.CONCLUSIONS

This work shows how we combine the conditional copula with marginal distributions for the GARCH innovations to build model to estimate the VAR and conduct the empirical studies in Nasdaq and S&P500 stock market. By using different copulas and marginal distributions for the GARCH innovations to estimate the VAR of a portfolio composed by Nasdaq and S&P500 stock indices from 2004-2014, we conclude that for α=0.05 the GARCH-t and Gumbel copula model provided a far better result in the VAR estimation; for α=0.1 the GARCH-N and Frank copula model was the best model. And GARCH-N prefer to model estimate lower risk while GARCH-t model prefer to estimate higher risk in market portfolio. On the other hand, we can see the relationship known as volatility clustering, in which large absolute return tend to follow large absolute returns and the same for small returns, between Nasdaq and S&P500 stock markets.

VI.ATTACHMENTS

A. References

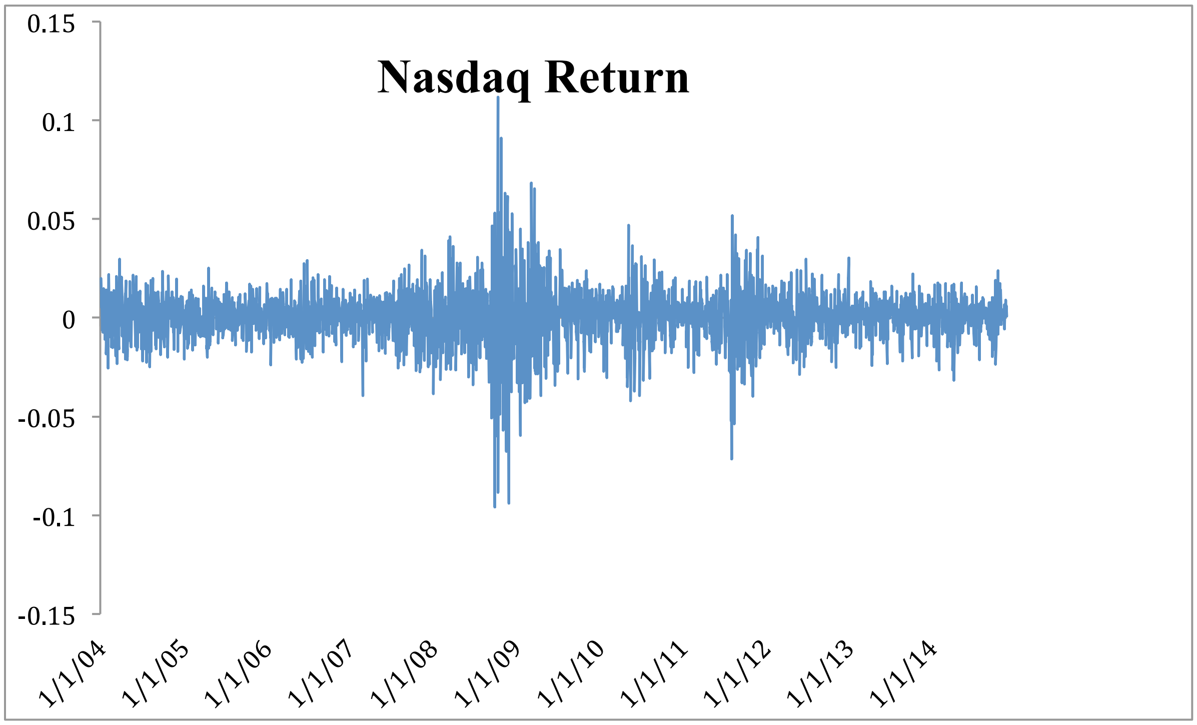
[1] Galbraith, J. W., & Zernov, S. (2009). Extreme dependence in the NASDAQ and S&P 500 composite indexes. *Applied Financial Economics*, *19*(13), 1019-1028.

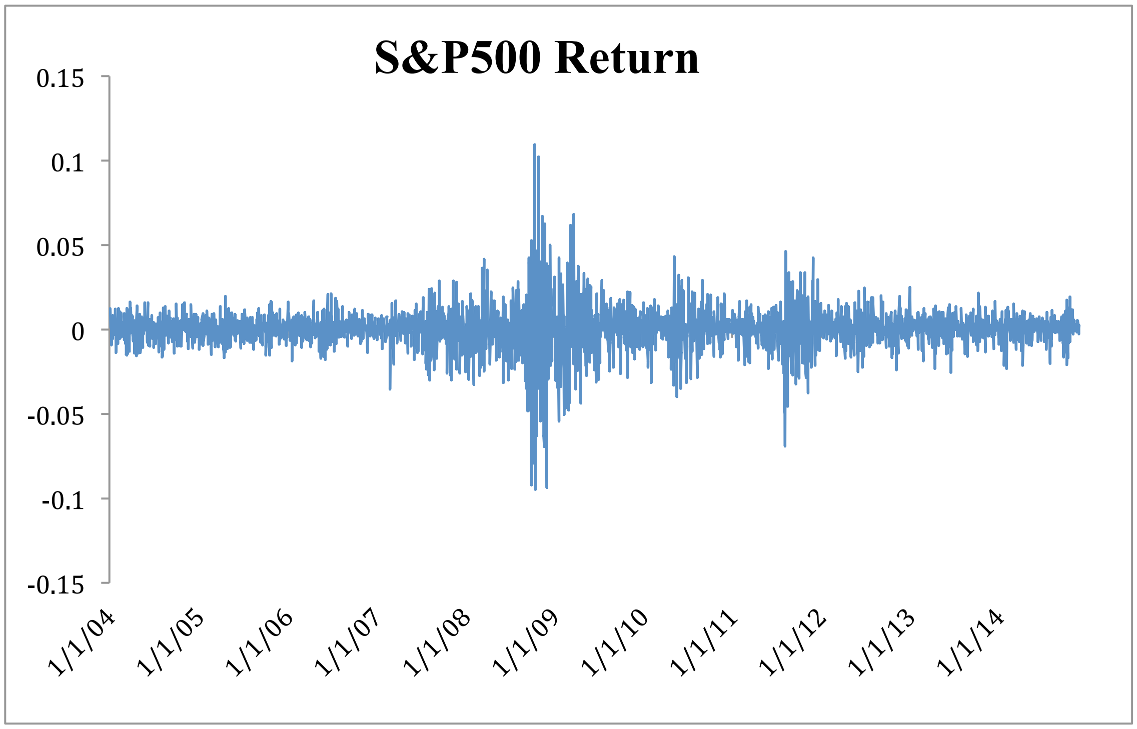
[2] He, X., & Gong, P. (2009). Measuring the coupled risks: A copula-based CVaR model. *Journal of Computational and Applied Mathematics*, *223*(2), 1066-1080.

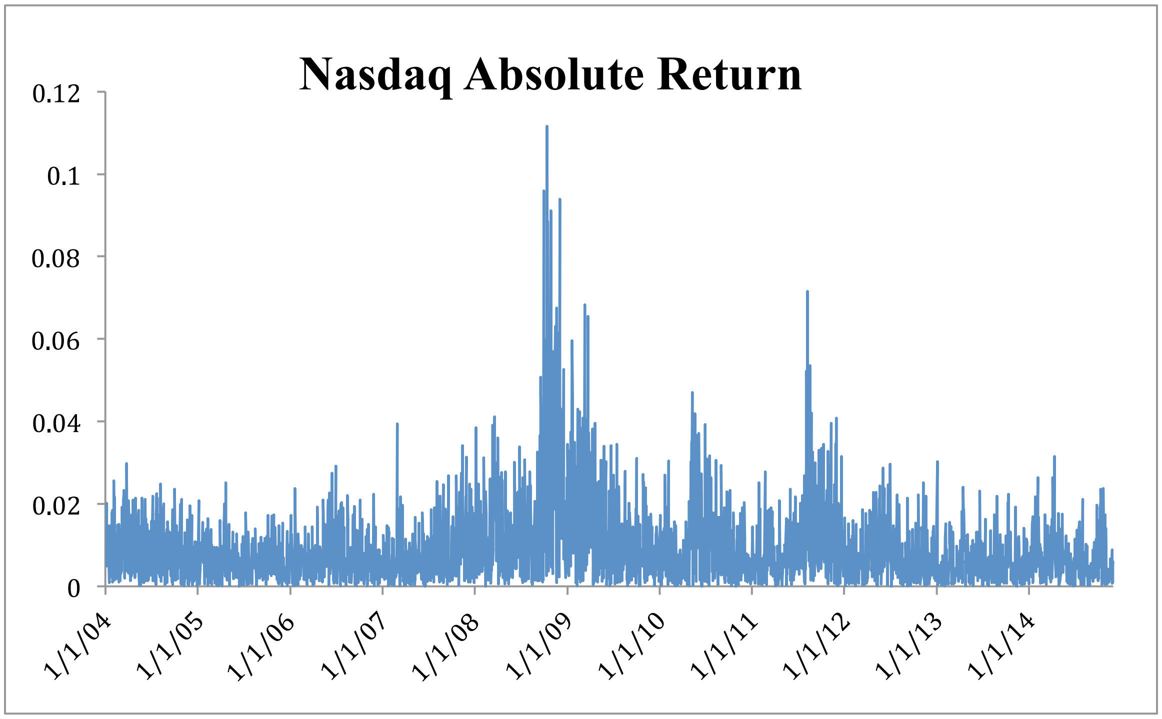
[3] Romano, C. (2002). Applying copula function to risk management. In *Capitalia, Italy. http://www. icer. it/workshop/Romano. pdf.*

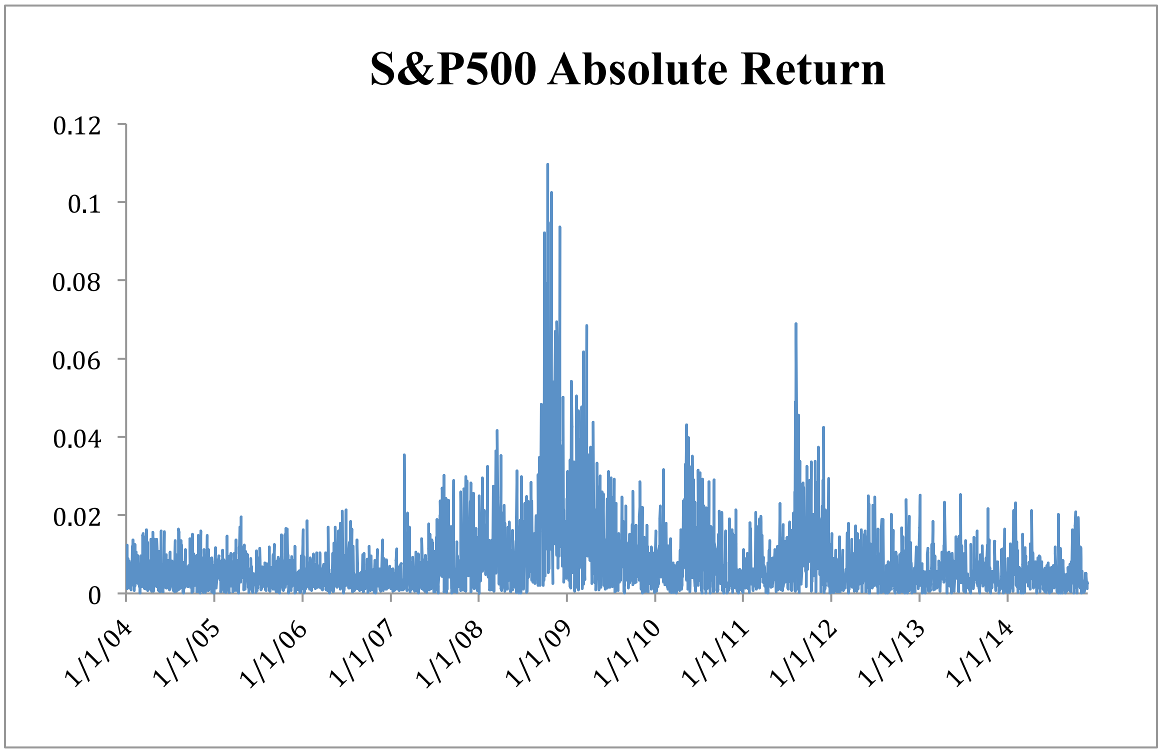
[4]Wang, Z. R., Chen, X. H., Jin, Y. B., & Zhou, Y. J. (2010). Estimating risk of foreign exchange portfolio: Using VaR and CVaR based on GARCH–EVT-Copula model. *Physica A: Statistical Mechanics and its Applications*, *389*(21), 4918-4928.

B. Data description




S&P500 Return
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
1/1/04
1/1/05
1/1/06
1/1/07
1/1/08
1/1/09
1/1/10
1/1/11
1/1/12
1/1/13
1/1/14


Nasdaq Absolute Return
0.12
0.1
0.08
0.06
0.04
0.02
0
1/1/04
1/1/05
1/1/06
1/1/07
1/1/08
1/1/09
1/1/10
1/1/11
1/1/12
1/1/13
1/1/14


S&P500 Absolute Return
0.12
0.1
0.08
0.06
0.04
0.02
0
1/1/04
1/1/05
1/1/06
1/1/07
1/1/08
1/1/09
1/1/10
1/1/11
1/1/12
1/1/13
1/1/14

C. R code for calculation

```
#############################################
## Estimating AR(1)GARCH(1,1) parameters
##############################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
   nasdaq = data[,4]
   sp500 = data[,5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

#############################################
## Estimating parameters of copulas
##############################################
library(VineCD)

##fit the Student-t Copula for GARCH-N
BiCopEst(nas.norm.cdf,sp500.norm.cdf,2,method="mle")
##fit the Student-t Copula for Garch-N
BiCopEst(nas.t.cdf,sp.t.cdf,2, method="mle")
##fit the Gumbel Copula for GARCH-N
BiCopEst(nas.norm.cdf,sp500.norm.cdf,4,method="mle")
##fit the Gumbel Copula for Garch-T
BiCopEst(nas.t.cdf,sp.t.cdf,4, method="mle")
##fit the Frank Copula for GARCH-N
BiCopEst(nas.norm.cdf,sp500.norm.cdf,5,method="mle")
```

```
##fit the Frank Copula for Garch-T
BiCopEst(nas.t.cdf,sp.t.cdf,5, method="mle")
##fit the Clayton Copula for GARCH-N
BiCopEst(nas.norm.cdf,sp500.norm.cdf,5,method="mle")
##fit the Clayton Copula for Garch-T
BiCopEst(nas.t.cdf,sp.t.cdf,3, method="mle")


###########################################################################
## comparing tail distances of estimated copulas with empitical copula
###########################################################################
## empirical margin distribution
sp500.e.cdf=order(sp500.n.resi)/2746
nas.e.cdf=order(nas.n.resi)/2746
##Student-t Copula for GARCH-E
BiCopEst(nas.e.cdf,sp500.e.cdf,2, method="mle")
##Clayton Copula for GARCH-E
BiCopEst(nas.e.cdf,sp500.e.cdf,3, method="mle")
##Gumbel Copula for GARCH-E
BiCopEst(nas.e.cdf,sp500.e.cdf,4, method="mle")
##Frank Copula for GARCH-E
BiCopEst(nas.e.cdf,sp500.e.cdf,5, method="mle")

#### to get table 5
install.packages("copBasic")
##calculate the empirical copula first
library(copBasic)
##EMRICcop(nas.t.resi,sp500.t.resi, para=NULL, bernstein=FALSE,
bernsteinprogress=TRUE)

##calculate the empirical copula using definition
##gamma=0.2
n=0
for(i in 1:2746)
{
  if ((nas.e.cdf[i]<=nas.e.cdf[550])&&(sp500.e.cdf[i]<=sp500.e.cdf[550]))
     n=n+1
}
em=n/2746
```

```
##clayton for GARCH N
BiCopCDF(0.2, 0.2, 3, 3.848723,par2=0.08688641)-em)^2
##clayton for GARCH T
BiCopCDF(0.2, 0.2, 3, 4.861411, par2=0.1014028)-em)^2
##student-t distribution for GARCH N
BiCopCDF(0.2, 0.2, 2, 14.91775,par2=3.429257)-em)^2
##student-t distribution for GARCH T
BiCopCDF(0.2, 0.2, 2, 8.479518,par2=1.593718)-em)^2

## Gumbel for garch N
(BiCopPDF(0.2, 0.2, 4, 4.314239,par2=0.06843129)-em)^2
##Gumbel for GARCH T
(BiCopCDF(0.2, 0.2, 4, 4.821897, par2=0.07717385)-em)^2
##Frank for Garch N
(BiCopCDF(0.2, 0.2, 5, 16.28931,par2=0.2918175)-em)^2
##Frank for GARCH T
(BiCopCDF(0.2, 0.2, 5,17.07474, par2=0.3038215)-em)^2

##gamma=0.1
n=0
for(i in 1:2746)
{
   if ((nas.e.cdf[i]<=nas.e.cdf[275])&&(sp500.e.cdf[i]<=sp500.e.cdf[275]))
      n=n+1
}
em=n/2746

##clayton for GARCH N
BiCopCDF(0.1, 0.1, 3, 3.848723,par2=0.08688641)-em)^2
##clayton for GARCH T
BiCopCDF(0.1, 0.1, 3, 4.861411, par2=0.1014028)-em)^2
##student-t distribution for GARCH N
BiCopCDF(0.1, 0.1, 2, 14.91775,par2=3.429257)-em)^2
##student-t distribution for GARCH T
BiCopCDF(0.1, 0.1, 2, 8.479518,par2=1.593718)-em)^2

## Gumbel for garch N
(BiCopPDF(0.1, 0.1, 4, 4.314239,par2=0.06843129)-em)^2
##Gumbel for GARCH T
(BiCopCDF(0.1, 0.1, 4, 4.821897, par2=0.07717385)-em)^2
##Frank for Garch N
```

```
(BiCopCDF(0.1, 0.1, 5, 16.28931,par2=0.2918175)-em)^2
##Frank for GARCH T
(BiCopCDF(0.1, 0.1, 5,17.07474, par2=0.3038215)-em)^2

##gamma=0.05
n=0
for(i in 1:2746)
{
   if ((nas.e.cdf[i]<=nas.e.cdf[137])&&(sp500.e.cdf[i]<=sp500.e.cdf[137]))
      n=n+1
}
em=n/2746

##clayton for GARCH N
BiCopCDF(0.05, 0.05, 3, 3.848723,par2=0.08688641)-em)^2
##clayton for GARCH T
BiCopCDF(0.05, 0.05, 3, 4.861411, par2=0.1014028)-em)^2
##student-t distribution for GARCH N
BiCopCDF(0.05, 0.05, 2, 14.91775,par2=3.429257)-em)^2
##student-t distribution for GARCH T
BiCopCDF(0.05, 0.05, 2, 8.479518,par2=1.593718)-em)^2

## Gumbel for garch N
(BiCopPDF(0.05, 0.05, 4, 4.314239,par2=0.06843129)-em)^2
##Gumbel for GARCH T
(BiCopCDF(0.05, 0.05, 4, 4.821897, par2=0.07717385)-em)^2
##Frank for Garch N
(BiCopCDF(0.05, 0.05, 5, 16.28931,par2=0.2918175)-em)^2
##Frank for GARCH T
(BiCopCDF(0.05, 0.05, 5,17.07474, par2=0.3038215)-em)^2

###############################################
## Computing VaR for estimated GARCH Plackett model
################################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
data = data[1:2700,]
```

```
window = seq(0,1900,50)
count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with    the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
   sp500.norm.cdf=pnorm(garch.n.nas.resid)
   ##calculate with t distribution using degree of freedom from the table before
   nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
   sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])



   ##fit the plackett Copula for GARCH-N
   plackett.n = BiCopEst(nas.norm.cdf,sp500.norm.cdf,family=4,method="mle",se=T)
   theta.n = plackett.n$par

   #generate 5000 innorvation terms from estimated copulas
   plackett.n.wn = rCopula(5000,plackettCopula(theta.n,dim=2))
   w = matrix(c(1/2,1/2),1,2)
   ## forecast mean and variance of garch to work out innavation vectors
   pred.n.nas.mat = predict(garch.n.nas,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(plackett.n.wn[,1])
   error = t(pred.n.nas.mat[,3]%*%t(wn))
   pred.n.nas = as.matrix(pred.n.nas.mat[,1])
   re = matrix(rep(pred.n.nas,each=5000),nrow=5000)##5000 by 50
   for.n.nas = re+error


   pred.n.sp.mat = predict(garch.n.sp,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(plackett.n.wn[,2])
   error = t(pred.n.sp.mat[,3]%*%t(wn))
   pred.n.sp = as.matrix(pred.n.sp.mat[,1])
   re = matrix(rep(pred.n.sp,each=5000),nrow=5000)##5000 by 50
   for.n.sp = re+error

   for(j in 1:50) {
      var1 = quantile(for.n.nas,.95)
      var2 = quantile(for.n.sp,.95)
      price.port = (c(var1,var2))%*%t(w)
      true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
      if (price.port < true.p){
         count = count + 1
      }
   }

}


count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
```

```
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with    the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
   sp500.norm.cdf=pnorm(garch.n.nas.resid)
   ##calculate with t distribution using degree of freedom from the table before
   nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
   sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])

   ##fit the plackett Copula for GARCH-N
   plackett.t = BiCopEst(nas.t.cdf,sp500.t.cdf,family=4,method="mle",se=T)
   theta.t = plackett.t$par

   #generate 5000 innorvation terms from estimated copulas
   plackett.t.wn = rCopula(5000,plackettCopula(theta.t,dim=2))

   w = matrix(c(1/2,1/2),1,2)
   ## forecast mean and variance of garch to work out innavation vectors
   pred.t.nas.mat = predict(garch.t.nas,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(plackett.t.wn[,1])
   error = t(pred.t.nas.mat[,3]%*%t(wn))
   pred.t.nas = as.matrix(pred.t.nas.mat[,1])
   re = matrix(rep(pred.t.nas,each=5000),nrow=5000)##5000 by 50
   for.t.nas = re+error

   pred.t.sp.mat = predict(garch.t.sp,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(plackett.t.wn[,2])
   error = t(pred.t.sp.mat[,3]%*%t(wn))
   pred.t.sp = as.matrix(pred.t.sp.mat[,1])
   re = matrix(rep(pred.t.sp,each=5000),nrow=5000)##5000 by 50
   for.t.sp = re+error

   for(j in 1:50) {
      var1 = quantile(for.t.nas,.95)
      var2 = quantile(for.t.sp,.95)
      price.port = (c(var1,var2))%*%t(w)
      true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
      if (price.port < true.p){
         count = count + 1
      }
   }

}

##############################################
## Computing VaR for estimated GARCH t-Copula model
###############################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
data = data[1:2700,]
window = seq(0,1900,50)
count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
```

```
coef(garch.n.nas)
coef(garch.t.nas)
coef(garch.n.sp)
coef(garch.t.sp)

garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

##caculate the CDF with   the standard normal distribution
nas.norm.cdf=pnorm(garch.n.sp.resid)
sp500.norm.cdf=pnorm(garch.n.nas.resid)
##calculate with t distribution using degree of freedom from the table before
nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])



##fit the std_t Copula for GARCH-N
std_t.n = BiCopEst(nas.norm.cdf,sp500.norm.cdf,family=4,method="mle",se=T)
theta.n = std_t.n$par

#generate 5000 innorvation terms from estimated copulas
std_t.n.wn = rCopula(5000,std_tCopula(theta.n,dim=2))

w = matrix(c(1/2,1/2),1,2)
## forecast mean and variance of garch to work out innavation vectors
pred.n.nas.mat = predict(garch.n.nas,n.ahead=50,mse="uncond",plot=T)
wn = qnorm(std_t.n.wn[,1])
error = t(pred.n.nas.mat[,3]%*%t(wn))
pred.n.nas = as.matrix(pred.n.nas.mat[,1])
re = matrix(rep(pred.n.nas,each=5000),nrow=5000)##5000 by 50
for.n.nas = re+error


pred.n.sp.mat = predict(garch.n.sp,n.ahead=50,mse="uncond",plot=T)
wn = qnorm(std_t.n.wn[,2])
error = t(pred.n.sp.mat[,3]%*%t(wn))
pred.n.sp = as.matrix(pred.n.sp.mat[,1])
```

```
  re = matrix(rep(pred.n.sp,each=5000),nrow=5000)##5000 by 50
  for.n.sp = re+error

  for(j in 1:50) {
    var1 = quantile(for.n.nas,.95)
    var2 = quantile(for.n.sp,.95)
    price.port = (c(var1,var2))%*%t(w)
    true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
    if (price.port < true.p){
      count = count + 1
    }
  }

}


count = 0

for (d in window){

  nasdaq = data[(d+1):(d+750),4]
  sp500 = data[(d+1):(d+750),5]
  garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
  garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
  garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
  garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
  coef(garch.n.nas)
  coef(garch.t.nas)
  coef(garch.n.sp)
  coef(garch.t.sp)

  garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
  garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

  garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
  garch.n.nas.resid = residuals(garch.n.nas, standardize=T)
```

```
    ##caculate the CDF with    the standard normal distribution
    nas.norm.cdf=pnorm(garch.n.sp.resid)
    sp500.norm.cdf=pnorm(garch.n.nas.resid)
    ##calculate with t distribution using degree of freedom from the table before
    nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
    sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])

    ##fit the std_t Copula for GARCH-N
    std_t.t = BiCopEst(nas.t.cdf,sp500.t.cdf,family=4,method="mle",se=T)
    theta.t = std_t.t$par

    #generate 5000 innorvation terms from estimated copulas
    std_t.t.wn = rCopula(5000,std_tCopula(theta.t,dim=2))

    w = matrix(c(1/2,1/2),1,2)
    ## forecast mean and variance of garch to work out innavation vectors
    pred.t.nas.mat = predict(garch.t.nas,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(std_t.t.wn[,1])
    error = t(pred.t.nas.mat[,3]%*%t(wn))
    pred.t.nas = as.matrix(pred.t.nas.mat[,1])
    re = matrix(rep(pred.t.nas,each=5000),nrow=5000)##5000 by 50
    for.t.nas = re+error

    pred.t.sp.mat = predict(garch.t.sp,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(std_t.t.wn[,2])
    error = t(pred.t.sp.mat[,3]%*%t(wn))
    pred.t.sp = as.matrix(pred.t.sp.mat[,1])
    re = matrix(rep(pred.t.sp,each=5000),nrow=5000)##5000 by 50
    for.t.sp = re+error

    for(j in 1:50) {
      var1 = quantile(for.t.nas,.95)
      var2 = quantile(for.t.sp,.95)
      price.port = (c(var1,var2))%*%t(w)
      true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
      if (price.port < true.p){
        count = count + 1
      }
    }

}
```

```
###############################################
## Computing VaR for estimated GARCH Clayton model
################################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
data = data[1:2700,]
window = seq(0,1900,50)
count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with   the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
```

```
sp500.norm.cdf=pnorm(garch.n.nas.resid)
##calculate with t distribution using degree of freedom from the table before
nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])



##fit the clayton Copula for GARCH-N
clayton.n = BiCopEst(nas.norm.cdf,sp500.norm.cdf,family=4,method="mle",se=T)
theta.n = clayton.n$par

#generate 5000 innorvation terms from estimated copulas
clayton.n.wn = rCopula(5000,claytonCopula(theta.n,dim=2))

w = matrix(c(1/2,1/2),1,2)
## forecast mean and variance of garch to work out innavation vectors
pred.n.nas.mat = predict(garch.n.nas,n.ahead=50,mse="uncond",plot=T)
wn = qnorm(clayton.n.wn[,1])
error = t(pred.n.nas.mat[,3]%*%t(wn))
pred.n.nas = as.matrix(pred.n.nas.mat[,1])
re = matrix(rep(pred.n.nas,each=5000),nrow=5000)##5000 by 50
for.n.nas = re+error


pred.n.sp.mat = predict(garch.n.sp,n.ahead=50,mse="uncond",plot=T)
wn = qnorm(clayton.n.wn[,2])
error = t(pred.n.sp.mat[,3]%*%t(wn))
pred.n.sp = as.matrix(pred.n.sp.mat[,1])
re = matrix(rep(pred.n.sp,each=5000),nrow=5000)##5000 by 50
for.n.sp = re+error

for(j in 1:50) {
   var1 = quantile(for.n.nas,.95)
   var2 = quantile(for.n.sp,.95)
   price.port = (c(var1,var2))%*%t(w)
   true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
   if (price.port < true.p){
      count = count + 1
   }
}
```

```
}


count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with    the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
   sp500.norm.cdf=pnorm(garch.n.nas.resid)
   ##calculate with t distribution using degree of freedom from the table before
   nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
   sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])

   ##fit the clayton Copula for GARCH-N
   clayton.t = BiCopEst(nas.t.cdf,sp500.t.cdf,family=4,method="mle",se=T)
   theta.t = clayton.t$par

   #generate 5000 innorvation terms from estimated copulas
   clayton.t.wn = rCopula(5000,claytonCopula(theta.t,dim=2))
```

```
    w = matrix(c(1/2,1/2),1,2)
    ## forecast mean and variance of garch to work out innavation vectors
    pred.t.nas.mat = predict(garch.t.nas,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(clayton.t.wn[,1])
    error = t(pred.t.nas.mat[,3]%*%t(wn))
    pred.t.nas = as.matrix(pred.t.nas.mat[,1])
    re = matrix(rep(pred.t.nas,each=5000),nrow=5000)##5000 by 50
    for.t.nas = re+error

    pred.t.sp.mat = predict(garch.t.sp,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(clayton.t.wn[,2])
    error = t(pred.t.sp.mat[,3]%*%t(wn))
    pred.t.sp = as.matrix(pred.t.sp.mat[,1])
    re = matrix(rep(pred.t.sp,each=5000),nrow=5000)##5000 by 50
    for.t.sp = re+error

    for(j in 1:50) {
        var1 = quantile(for.t.nas,.95)
        var2 = quantile(for.t.sp,.95)
        price.port = (c(var1,var2))%*%t(w)
        true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
        if (price.port < true.p){
            count = count + 1
        }
    }

}

##############################################
## Computing VaR for estimated GARCH Gumbel model
###############################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
data = data[1:2700,]
window = seq(0,1900,50)
count = 0
```

```
for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with    the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
   sp500.norm.cdf=pnorm(garch.n.nas.resid)
   ##calculate with t distribution using degree of freedom from the table before
   nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
   sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])



   ##fit the Gumbel Copula for GARCH-N
   gumbel.n = BiCopEst(nas.norm.cdf,sp500.norm.cdf,family=4,method="mle",se=T)
   theta.n = gumbel.n$par

   #generate 5000 innorvation terms from estimated copulas
   gumbel.n.wn = rCopula(5000,gumbelCopula(theta.n,dim=2))

   w = matrix(c(1/2,1/2),1,2)
   ## forecast mean and variance of garch to work out innavation vectors
   pred.n.nas.mat = predict(garch.n.nas,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(gumbel.n.wn[,1])
   error = t(pred.n.nas.mat[,3]%*%t(wn))
   pred.n.nas = as.matrix(pred.n.nas.mat[,1])
   re = matrix(rep(pred.n.nas,each=5000),nrow=5000)##5000 by 50
   for.n.nas = re+error


   pred.n.sp.mat = predict(garch.n.sp,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(gumbel.n.wn[,2])
   error = t(pred.n.sp.mat[,3]%*%t(wn))
   pred.n.sp = as.matrix(pred.n.sp.mat[,1])
   re = matrix(rep(pred.n.sp,each=5000),nrow=5000)##5000 by 50
   for.n.sp = re+error

   for(j in 1:50) {
      var1 = quantile(for.n.nas,.95)
      var2 = quantile(for.n.sp,.95)
      price.port = (c(var1,var2))%*%t(w)
      true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
      if (price.port < true.p){
         count = count + 1
      }
   }

}


count = 0

for (d in window){

   nasdaq = data[(d+1):(d+750),4]
   sp500 = data[(d+1):(d+750),5]
   garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
   garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
   garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
```

```
   garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
   coef(garch.n.nas)
   coef(garch.t.nas)
   coef(garch.n.sp)
   coef(garch.t.sp)

   garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
   garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

   garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
   garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

   ##caculate the CDF with   the standard normal distribution
   nas.norm.cdf=pnorm(garch.n.sp.resid)
   sp500.norm.cdf=pnorm(garch.n.nas.resid)
   ##calculate with t distribution using degree of freedom from the table before
   nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
   sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])

   ##fit the Gumbel Copula for GARCH-N
   gumbel.t = BiCopEst(nas.t.cdf,sp500.t.cdf,family=4,method="mle",se=T)
   theta.t = gumbel.t$par

   #generate 5000 innorvation terms from estimated copulas
   gumbel.t.wn = rCopula(5000,gumbelCopula(theta.t,dim=2))

   w = matrix(c(1/2,1/2),1,2)
   ## forecast mean and variance of garch to work out innavation vectors
   pred.t.nas.mat = predict(garch.t.nas,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(gumbel.t.wn[,1])
   error = t(pred.t.nas.mat[,3]%*%t(wn))
   pred.t.nas = as.matrix(pred.t.nas.mat[,1])
   re = matrix(rep(pred.t.nas,each=5000),nrow=5000)##5000 by 50
   for.t.nas = re+error

   pred.t.sp.mat = predict(garch.t.sp,n.ahead=50,mse="uncond",plot=T)
   wn = qnorm(gumbel.t.wn[,2])
   error = t(pred.t.sp.mat[,3]%*%t(wn))
   pred.t.sp = as.matrix(pred.t.sp.mat[,1])
   re = matrix(rep(pred.t.sp,each=5000),nrow=5000)##5000 by 50
```

```
  for.t.sp = re+error

  for(j in 1:50) {
    var1 = quantile(for.t.nas,.95)
    var2 = quantile(for.t.sp,.95)
    price.port = (c(var1,var2))%*%t(w)
    true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
    if (price.port < true.p){
      count = count + 1
    }
  }

}

##############################################
## Computing VaR for estimated GARCH Frank model
###############################################
library(fGarch)
library(copBasic)
library(copula)
library(CDVine)

data = read.csv("returns.csv")
data = data[1:2700,]
window = seq(0,1900,50)
count = 0

for (d in window){

  nasdaq = data[(d+1):(d+750),4]
  sp500 = data[(d+1):(d+750),5]
  garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
  garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
  garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
  garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
  coef(garch.n.nas)
  coef(garch.t.nas)
  coef(garch.n.sp)
  coef(garch.t.sp)

  garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
  garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

  garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
  garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

  ##caculate the CDF with    the standard normal distribution
  nas.norm.cdf=pnorm(garch.n.sp.resid)
  sp500.norm.cdf=pnorm(garch.n.nas.resid)
  ##calculate with t distribution using degree of freedom from the table before
  nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
  sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])


  ##fit the Frank Copula for Garch-N
  frank.n = BiCopEst(nas.n.cdf,sp.n.cdf,family=5, method="mle",se=T)
  theta.n = frank.n$par

  #generate 5000 innorvation terms from estimated copulas
  frank.n.wn = rCopula(5000,frankCopula(theta.n,dim=2))

    w = matrix(c(1/2,1/2),1,2)
    ## forecast mean and variance of garch to work out innavation vectors
    pred.n.nas.mat = predict(garch.n.nas,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(frank.n.wn[,1])
    error = t(pred.n.nas.mat[,3]%*%t(wn))
    pred.n.nas = as.matrix(pred.n.nas.mat[,1])
    re = matrix(rep(pred.n.nas,each=5000),nrow=5000)##5000 by 50
    for.n.nas = re+error


    pred.n.sp.mat = predict(garch.n.sp,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(frank.n.wn[,2])
    error = t(pred.n.sp.mat[,3]%*%t(wn))
    pred.n.sp = as.matrix(pred.n.sp.mat[,1])
    re = matrix(rep(pred.n.sp,each=5000),nrow=5000)##5000 by 50
    for.n.sp = re+error
```

```
    for(j in 1:50) {
        var1 = quantile(for.n.nas,.95)
        var2 = quantile(for.n.sp,.95)
        price.port = (c(var1,var2))%*%t(w)
        true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
        if (price.port < true.p){
            count = count + 1
        }
    }

}


count = 0
for (d in window){

    nasdaq = data[(d+1):(d+750),4]
    sp500 = data[(d+1):(d+750),5]
    garch.n.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="norm",trace=F)
    garch.t.nas = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(nasdaq),cond.dist="std",trace=F)
    garch.n.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="norm",trace=F)
    garch.t.sp = garchFit(formula = ~ arma(1,0)+garch(1,1), data =
as.ts(sp500),cond.dist="std",trace=F)
    coef(garch.n.nas)
    coef(garch.t.nas)
    coef(garch.n.sp)
    coef(garch.t.sp)

    garch.t.sp.resid = residuals(garch.t.sp, standardize=T)
    garch.t.nas.resid = residuals(garch.t.nas, standardize=T)

    garch.n.sp.resid = residuals(garch.n.sp, standardize=T)
    garch.n.nas.resid = residuals(garch.n.nas, standardize=T)

    ##caculate the CDF with    the standard normal distribution
    nas.norm.cdf=pnorm(garch.n.sp.resid)
    sp500.norm.cdf=pnorm(garch.n.nas.resid)
    ##calculate with t distribution using degree of freedom from the table before
    nas.t.cdf=pt(garch.t.nas.resid,coef(garch.t.nas)[6])
    sp.t.cdf=pt(garch.t.sp.resid,coef(garch.t.sp)[6])


    ##fit the Frank Copula for Garch-t
    frank.t = BiCopEst(nas.t.cdf,sp.t.cdf,family=5, method="mle",se=T)
    theta.t = frank.t$par

    #generate 5000 innorvation terms from estimated copulas
    frank.t.wn = rCopula(5000,frankCopula(theta.t,dim=2))

    w = matrix(c(1/2,1/2),1,2)
    ## forecast mean and variance of garch to work out innavation vectors
    pred.t.nas.mat = predict(garch.t.nas,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(frank.t.wn[,1])
    error = t(pred.t.nas.mat[,3]%*%t(wn))
    pred.t.nas = as.matrix(pred.t.nas.mat[,1])
    re = matrix(rep(pred.t.nas,each=5000),nrow=5000)##5000 by 50
    for.t.nas = re+error


    pred.t.sp.mat = predict(garch.t.sp,n.ahead=50,mse="uncond",plot=T)
    wn = qnorm(frank.t.wn[,2])
    error = t(pred.t.sp.mat[,3]%*%t(wn))
    pred.t.sp = as.matrix(pred.t.sp.mat[,1])
    re = matrix(rep(pred.t.sp,each=5000),nrow=5000)##5000 by 50
    for.t.sp = re+error

    for(j in 1:50) {
      var1 = quantile(for.t.nas,.95)
      var2 = quantile(for.t.sp,.95)
      price.port = (c(var1,var2))%*%t(w)
      true.p = (c(data[(d+750+j),4],data[(d+750+j),5]))%*%t(w)
      if (price.port < true.p){
        count = count + 1
      }
    }

  }
```