

UW

W

COMPUTATIONAL FINANCE & RISK MANAGEMENT

UNIVERSITY of WASHINGTON

Department of Applied Mathematics

Estimating Risk Measures

Amath 546/Econ 589

Eric Zivot

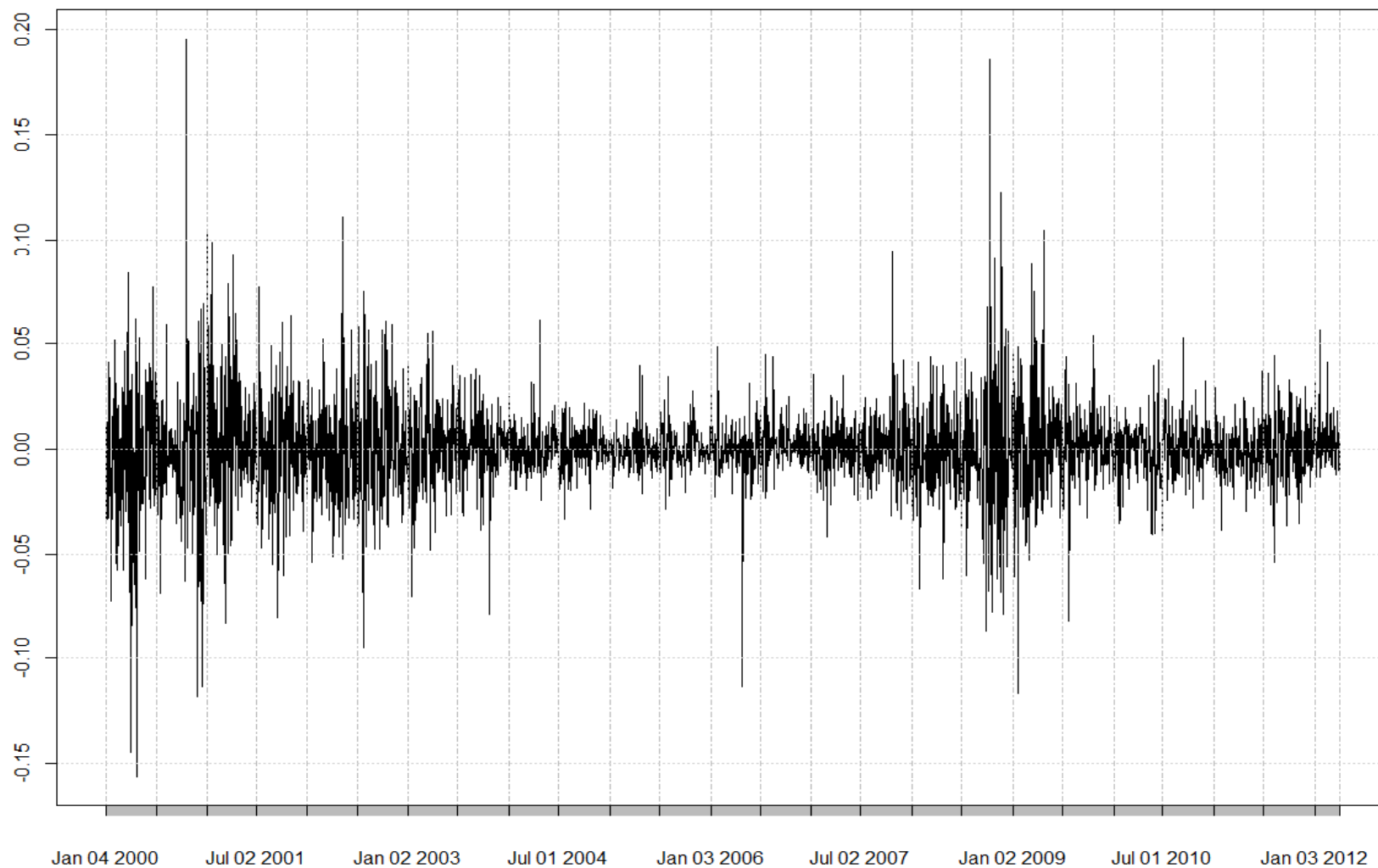
Spring 2013

Updated: April 23, 2013

© Eric Zivot 2008

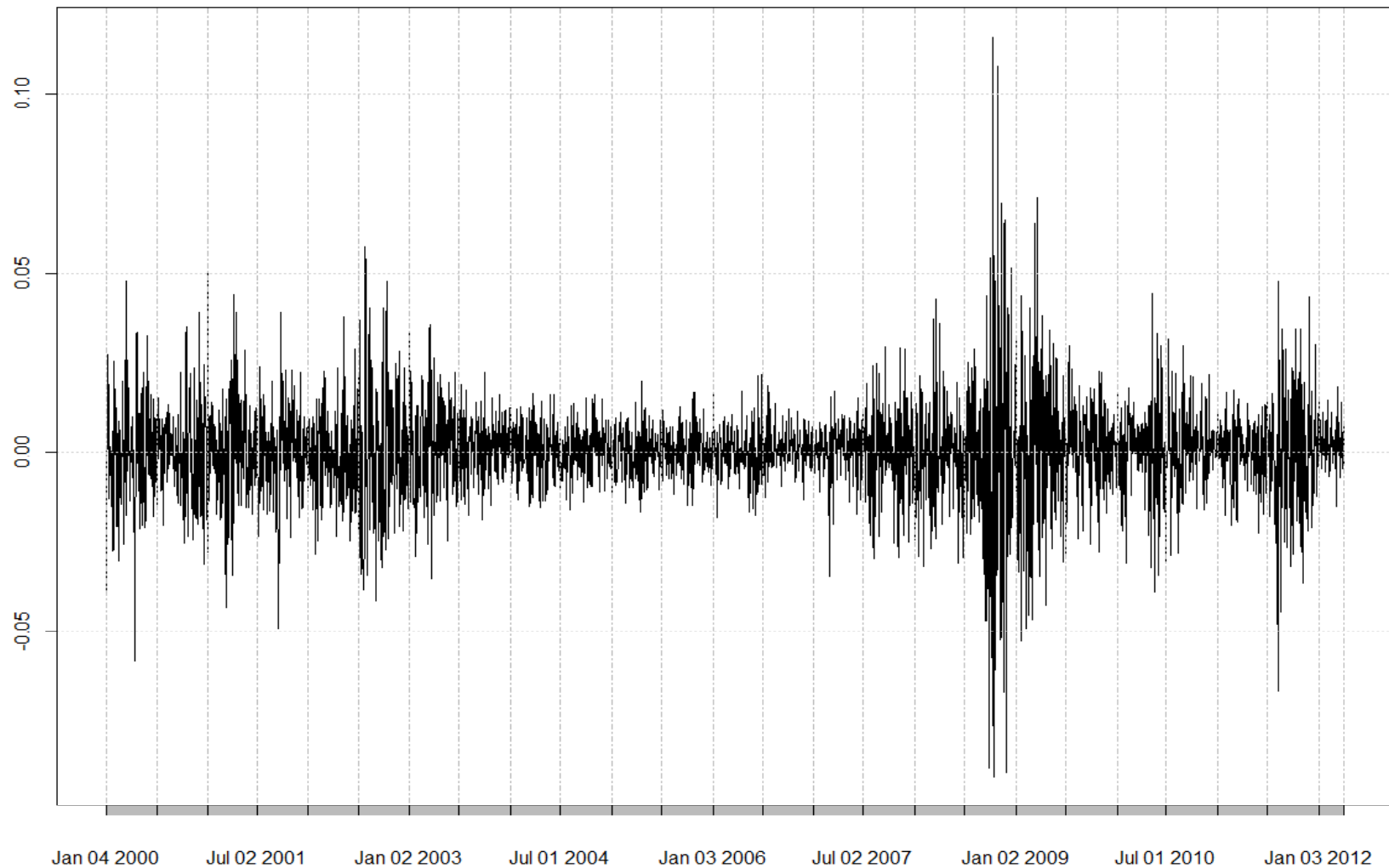
Daily Simple Returns on MSFT

MSFT.ret



Daily Simple Returns on S&P 500

GSPC.ret



Nonparametric Risk Measures

```
# combined data
```

```
> MSFT.GSPC.ret = cbind(MSFT.ret,GSPC.ret)
```

```
# standard deviation
```

```
> apply(MSFT.GSPC.ret, 2, sd)
```

```
MSFT.Adjusted  GSPC.Adjusted
      0.02139      0.01374
```

```
# empirical 5% and 1% quantiles (historical VaR)
```

```
> apply(MSFT.GSPC.ret, 2, quantile, probs=c(0.05, 0.01))
```

```
MSFT.Adjusted  GSPC.Adjusted
5%      -0.03116      -0.02138
1%      -0.06058      -0.03838
```

Nonparametric Risk Measures

```
# function to compute historical 5% and 1% ES
ES.fun = function(x, alpha=0.05) {
  qhat = quantile(x, probs=alpha)
  mean(x[x <= qhat])
}
```

```
> apply(MSFT.GSPC.ret, 2, ES.fun, alpha=0.05)
MSFT.Adjusted GSPC.Adjusted
-0.04970      -0.03256
```

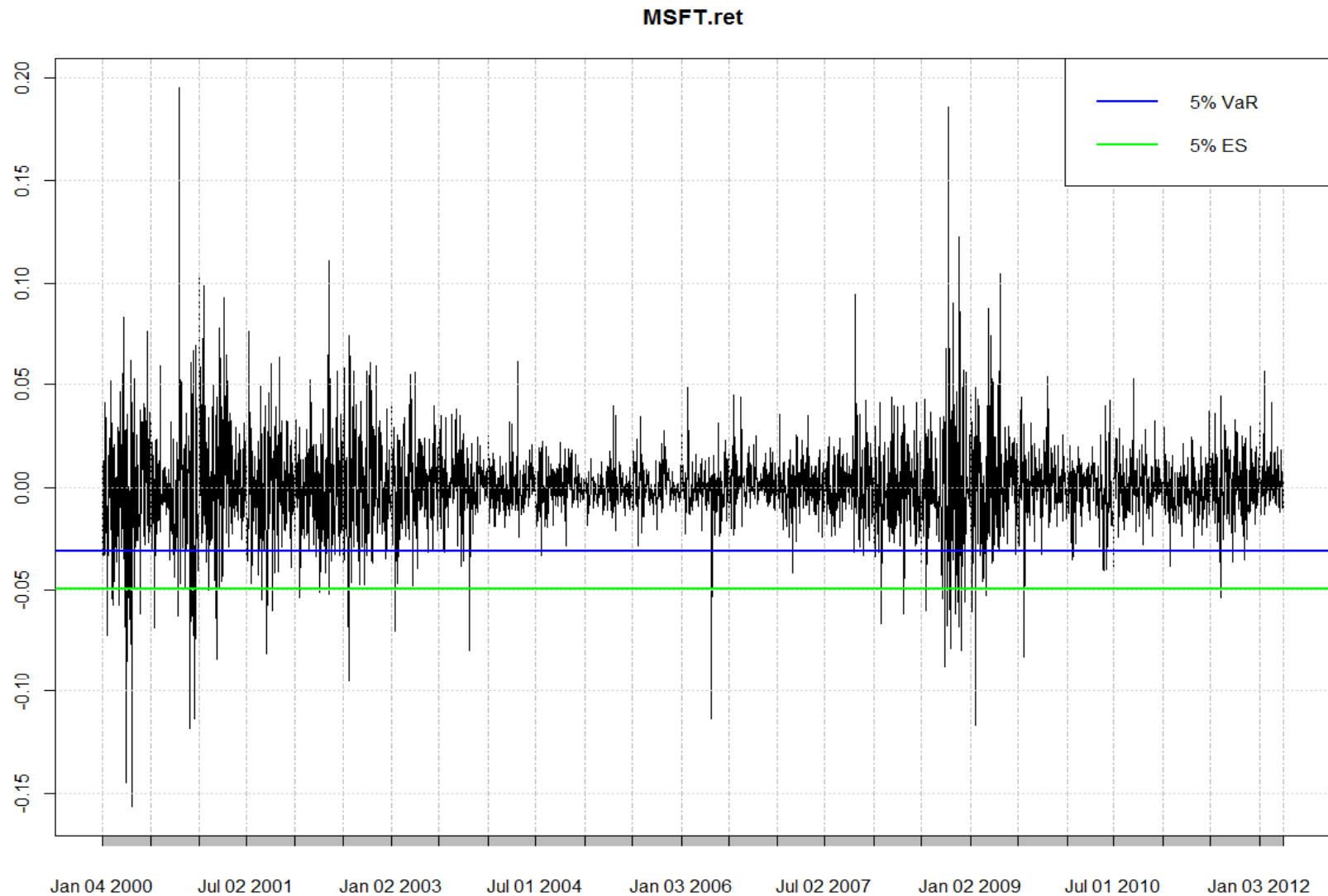
```
> apply(MSFT.GSPC.ret, 2, ES.fun, alpha=0.01)
MSFT.Adjusted GSPC.Adjusted
-0.08252      -0.05400
```

PerformanceAnalytics Functions

```
> library(PerformanceAnalytics)
> VaR(MSFT.GSPC.ret, p=0.95, method="historical")
      MSFT.Adjusted  GSPC.Adjusted
VaR      -0.03116      -0.02138

> ES(MSFT.GSPC.ret, p=0.95, method="historical")
      MSFT.Adjusted  GSPC.Adjusted
ES      -0.0497      -0.03256
```

Nonparametric 5% VaR and ES



Normal VaR and ES

```
# MLEs for mu and sigma
```

```
> mu.hat = apply(MSFT.GSPC.ret, 2, mean)
```

```
> sigma.hat = apply(MSFT.GSPC.ret, 2, sd)
```

```
# normal quantile and ES estimates
```

```
> q.05.z = mu.hat + sigma.hat*qnorm(0.05)
```

```
> q.01.z = mu.hat + sigma.hat*qnorm(0.01)
```

```
# mean return below quantile estimates
```

```
> es.05.norm = -(mu.hat + sigma.hat*dnorm(qnorm(0.05))/0.05)
```

```
> es.01.norm = -(mu.hat + sigma.hat*dnorm(qnorm(0.01))/0.01)
```


Normal VaR and ES

```
> mu.hat
```

```
MSFT.Adjusted  GSPC.Adjusted  
1.161e-04      8.497e-05
```

```
> sigma.hat
```

```
MSFT.Adjusted  GSPC.Adjusted  
0.02139       0.01374
```

```
> q.05.norm
```

```
MSFT.Adjusted  GSPC.Adjusted  
-0.03507      -0.02252
```

```
> q.01.norm
```

```
MSFT.Adjusted  GSPC.Adjusted  
-0.04965      -0.03188
```

```
> es.05.norm
```

```
MSFT.Adjusted  GSPC.Adjusted  
-0.04424      -0.02843
```

```
> es.01.norm
```

```
MSFT.Adjusted  GSPC.Adjusted  
-0.05713      -0.03671
```

PerformanceAnalytics Functions

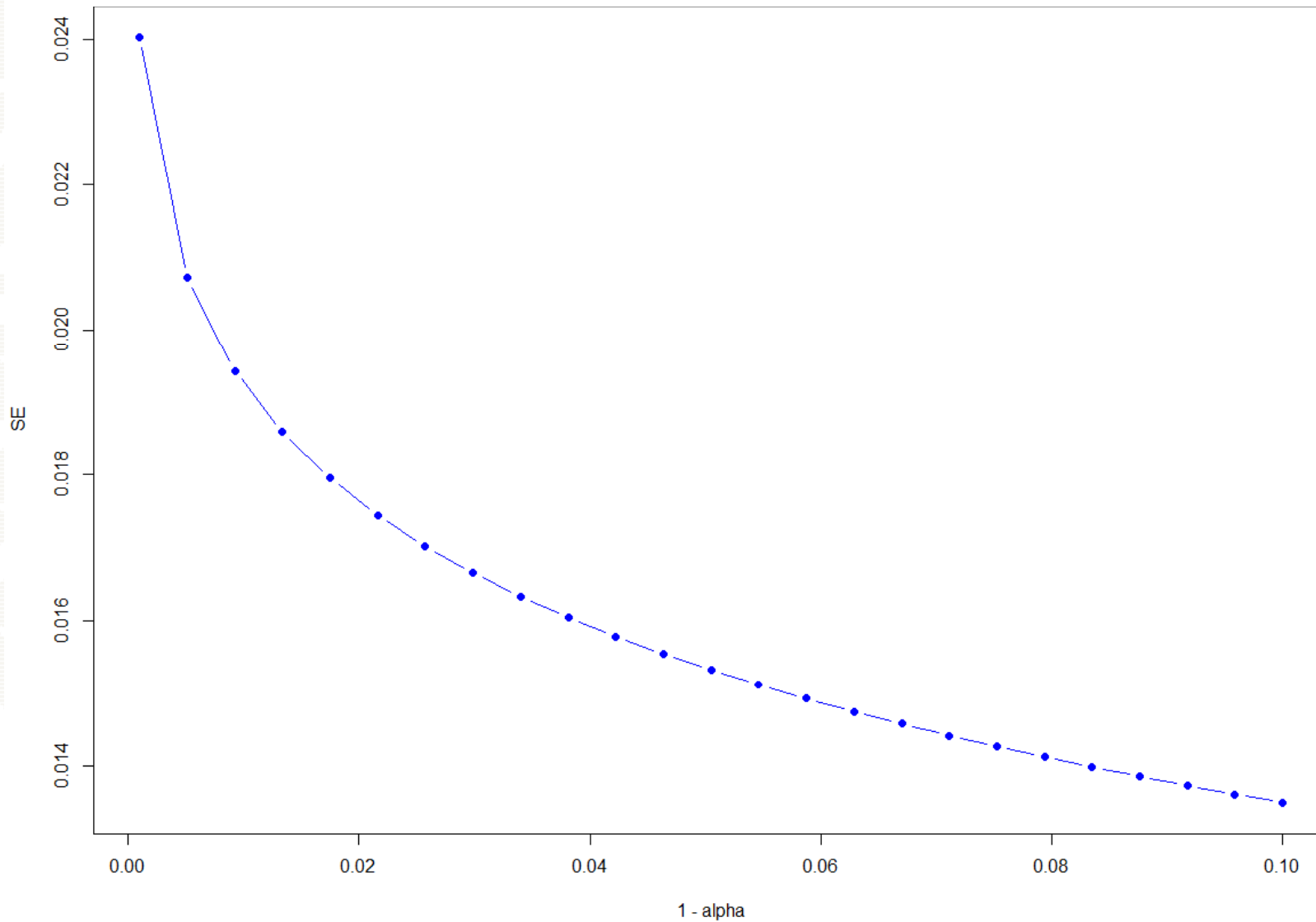
```
> VaR(MSFT.GSPC.ret, p=0.95, method="gaussian")
```

	MSFT.Adjusted	GSPC.Adjusted
VaR	-0.03506	-0.02251

```
> ES(MSFT.GSPC.ret, p=0.95, method="gaussian")
```

	MSFT.Adjusted	GSPC.Adjusted
ES	-0.044	-0.02826

Standard Error of Normal Quantile



MLE of Student's t Distribution

```
# Fit Student's t distribution by MLE using fitdistr from
# MASS package
# Assume MSFT.ret is student's t with parameters mu, sigma
# and v
# note: E[MSFT.ret] = mu,
# var(MSFT.ret) = sigma^2 * (v/(v-2))
# scale returns by 100 to improve numerical stability of MLE
> library(MASS)
> MSFT.t.mle = fitdistr(MSFT.ret*100, densfun="t")
> MSFT.t.mle
```

m	s	df
-0.006685	1.250321	2.659188
(0.027828)	(0.032016)	(0.157775)

```
# extract and rescale estimates
> theta.hat = coef(MSFT.t.mle)
> mu.MSFT.t = theta.hat["m"]/100
> sigma.MSFT.t = theta.hat["s"]/100
> v.MSFT.t = theta.hat["df"]
```

Student's t VaR for MSFT

```
# Standard t quantiles
> q.t.05 = qt(0.05, df=v.MSFT.t)
> q.t.01 = qt(0.01, df=v.MSFT.t)

# Estimated t Quantiles for MSFT
> q.MSFT.t.05 = mu.MSFT.t + sigma.MSFT.t*q.t.05
> q.MSFT.t.01 = mu.MSFT.t + sigma.MSFT.t*q.t.01

> q.MSFT.t.05
      m
-0.03109

> q.MSFT.t.01
      m
-0.06309
```

Student's t ES for MSFT

```
# Student-t multipliers for ES
> t.adj.05 = (dt(q.t.05, df=v.MSFT.t)/0.05)*((v.MSFT.t +
+      q.t.05^2)/(v.MSFT.t - 1))
> t.adj.01 = (dt(q.t.01, df=v.MSFT.t)/0.01)*((v.MSFT.t +
+      q.t.01^2)/(v.MSFT.t - 1))
# Student t ES estimates
> es.MSFT.t.05 = -(mu.MSFT.t + sigma.MSFT.t*t.adj.05)
> es.MSFT.t.01 = -(mu.MSFT.t + sigma.MSFT.t*t.adj.01)

> es.MSFT.t.05
      m
-0.05388

> es.MSFT.t.01
      m
-0.1032
```

Simulating Student's t VaR and ES

```
# simulate data from fitted Student's t distribution
# t.v ~ standardized Student t with v df. E[t.v] = 0,
# var(t.v) = v/(v-2)
> set.seed(123)
> t.sim = mu.MSFT.t + sigma.MSFT.t*rt(n=10000,
+                                     df=v.MSFT.t)
> q.t.sim = quantile(t.sim, probs=c(0.05, 0.01))
> es.t.05 = mean(t.sim[t.sim <= q.t.sim[1]])
> es.t.01 = mean(t.sim[t.sim <= q.t.sim[2]])

# simulated quantiles
> q.t.sim
      5%      1%
-0.03253 -0.06643

# simulated ES
> es.t.05
[1] -0.05563
> es.t.01
[1] -0.1025
```

Modified VaR and ES

```
# Use PerformanceAnalytics Functions VaR and ES

# Modified VaR
> modifiedVaR.05 = VaR(MSFT.GSPC.ret, p=0.95,
+                       method="modified")
> modifiedVaR.01 = VaR(MSFT.GSPC.ret, p=0.99,
+                       method="modified")

# Modified ES
> modifiedES.05 = ES(MSFT.GSPC.ret, p=0.95,
+                   method="modified")
> modifiedES.01 = ES(MSFT.GSPC.ret, p=0.99,
+                   method="modified")
```


Modified VaR and ES

```
> modifiedVaR.05
```

```
MSFT.Adjusted GSPC.Adjusted  
VaR          -0.02962      -0.02036
```

```
> modifiedVaR.01
```

```
MSFT.Adjusted GSPC.Adjusted  
VaR          -0.09036      -0.05513
```

```
> modifiedES.05
```

```
MSFT.Adjusted GSPC.Adjusted  
ES           -0.02962      -0.02974
```

```
> modifiedES.01
```

```
MSFT.Adjusted GSPC.Adjusted  
ES           -0.09036      -0.05513
```

Note: results are strange – VaR and ES estimates are the same !!!

Estimating Portfolio Risk Measures

```
# equally weighted portfolio of MSFT and GSPC
> port.ret = 0.5*MSFT.ret + 0.5*GSPC.ret
> colnames(port.ret) = "port"

> mean(port.ret)
[1] 0.0001005

> sd(port.ret)
  port
0.01615
Warning message:
sd(<matrix>) is deprecated.
Use apply(*, 2, sd) instead.

> sd(as.numeric(port.ret))
[1] 0.01615
```

Estimating Portfolio Risk Measures

```
# Estimating portfolio vol using covariance matrix and
# weights
> Sigma.hat = cov(MSFT.GSPC.ret)
> w = c(0.5, 0.5)
> sqrt(t(w)%*%Sigma.hat%*%w)
      [,1]
[1,] 0.01615
```

Estimating Volatility Risk Budgets

```
# estimating contributions to Vol
> sigma.p.hat = as.numeric(sqrt(t(w)%*%Sigma.hat%*%w))
> mcr.vol = Sigma.hat %*% w / sigma.p.hat
> cr.vol = w * mcr.vol
> pcr.vol = cr.vol / sigma.p.hat
> cbind(mcr.vol, cr.vol, pcr.vol)
      [,1]      [,2]      [,3]
MSFT.Adjusted 0.02031 0.010154 0.6289
GSPC.Adjusted 0.01199 0.005993 0.3711

# check
> sigma.p.hat
[1] 0.01615
> sum(cr.vol)
[1] 0.01615
> sum(pcr.vol)
[1] 1
```

Estimating Volatility Risk Budgets

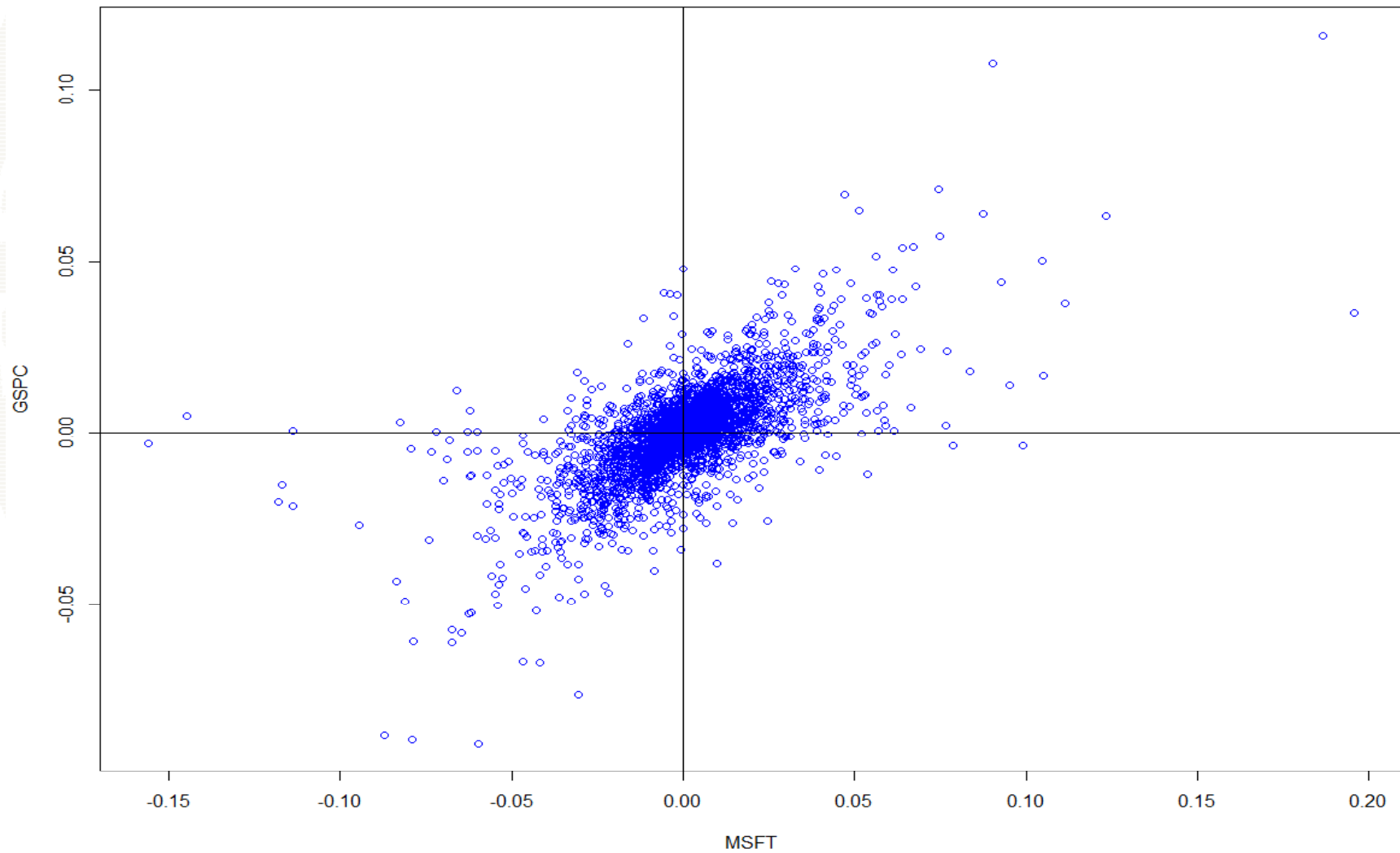
```
# Use PerformanceAnalytics function StdDev()
> StdDev(MSFT.GSPC.ret, portfolio_method="component",
+ weights=c(0.5, 0.5))
$StdDev
      [,1]
[1,] 0.01615

$contribution
[1] 0.010154 0.005993

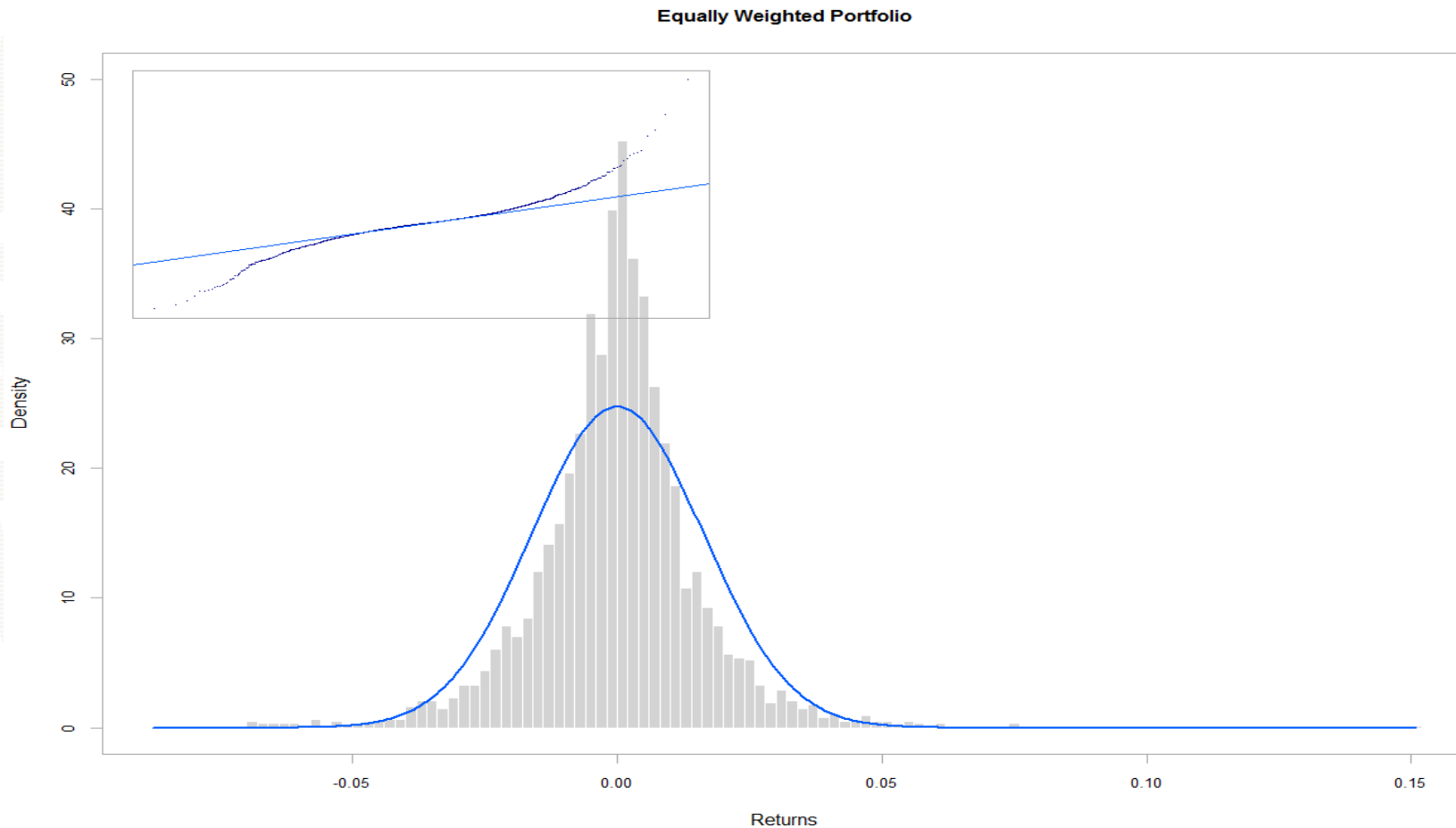
$pct_contrib_StdDev
[1] 0.6289 0.3711
```

Empirical Bivariate Distribution

Empirical Bivariate Distribution of Returns



Empirical Portfolio Return Distribution



```
> chart.Histogram(port.ret, main="Equally Weighted Portfolio",  
+ methods=c("add.normal","add.qqplot"))
```

Nonparametric Estimation of Portfolio VaR and ES

```
# VaR
> VaR(port.ret, p=0.95, method="historical")
      port
VaR -0.02514
> VaR(port.ret, p=0.99, method="historical")
      port
VaR -0.04445

# ES
> ES(port.ret, p=0.95, method="historical")
      port
ES -0.03761
> ES(port.ret, p=0.99, method="historical")
      port
ES -0.06024
```


Bivariate Normal Distribution

```
> mu.hat = apply(MSFT.GSPC.ret, 2, mean)
> Sigma.hat = cov(MSFT.GSPC.ret)
> Cor.hat = cov2cor(Sigma.hat)
```

```
> mu.hat
```

	MSFT.Adjusted	GSPC.Adjusted
	1.161e-04	8.497e-05

```
> Sigma.hat
```

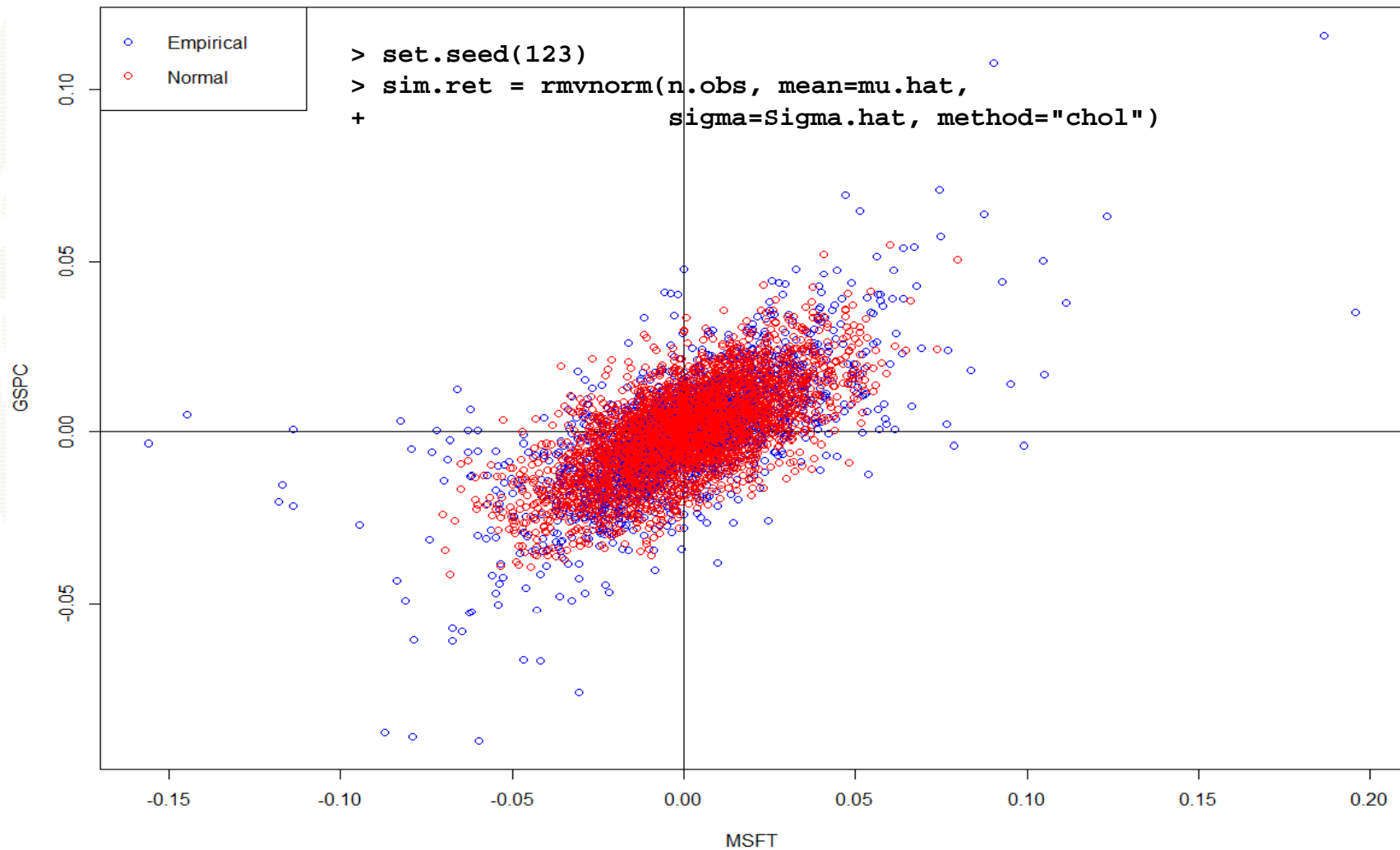
	MSFT.Adjusted	GSPC.Adjusted
MSFT.Adjusted	0.0004576	0.0001982
GSPC.Adjusted	0.0001982	0.0001888

```
> Cor.hat
```

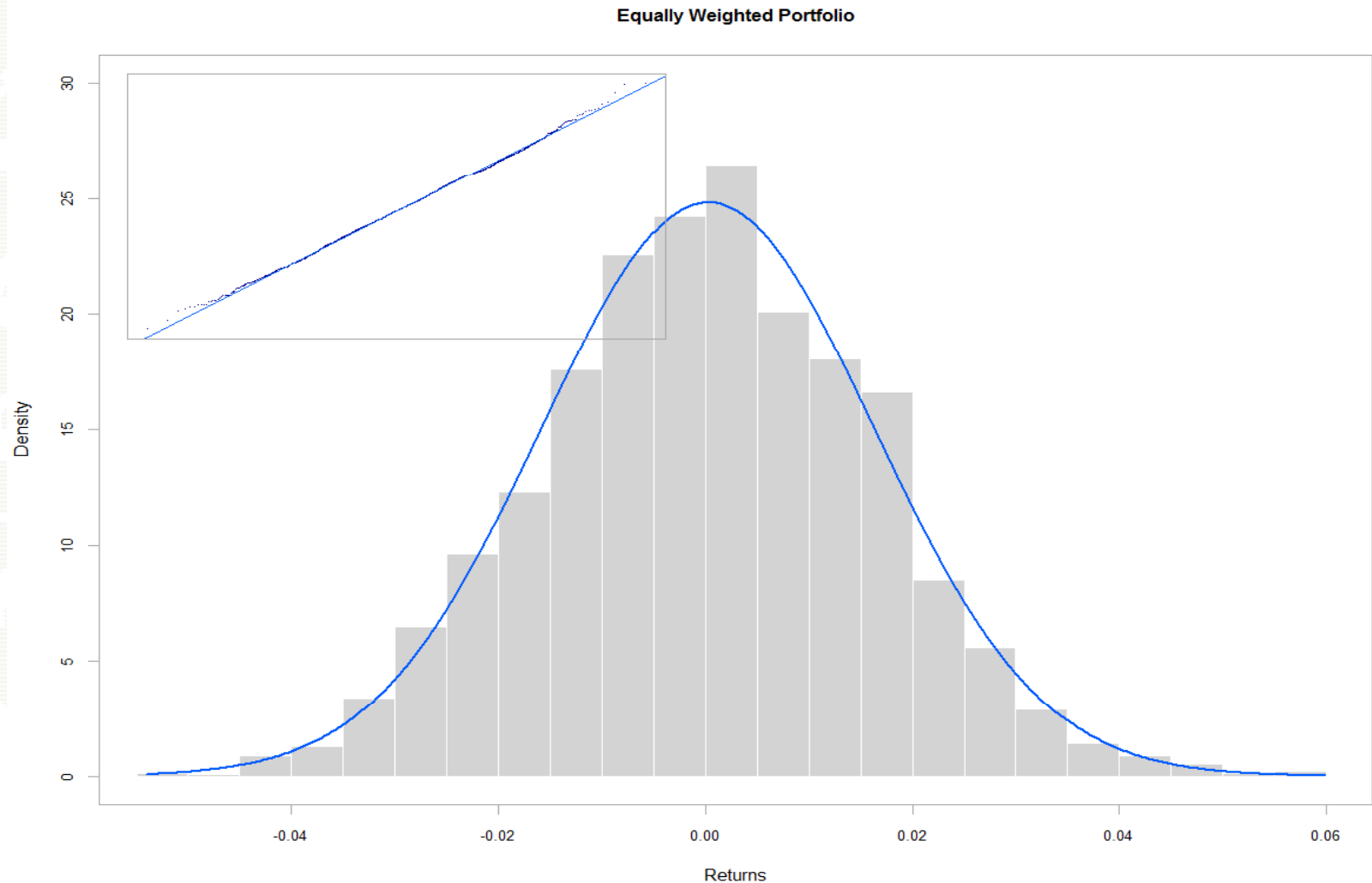
	MSFT.Adjusted	GSPC.Adjusted
MSFT.Adjusted	1.0000	0.6743
GSPC.Adjusted	0.6743	1.0000

Simulated Bivariate Normal Returns

Empirical vs. Bivariate Normal



Simulated Normal Portfolio Returns



```
> port.ret.sim = 0.5*sim.ret[,1] + 0.5*sim.ret[,2]
```

Estimated Portfolio Risk Measures

```
# volatility
> StdDev(port.ret)
      [,1]
StdDev 0.01615

# VaR
> VaR(port.ret, p = 0.95, method="gaussian")
      port
VaR -0.02645
> VaR(port.ret, p = 0.99, method="gaussian")
      port
VaR -0.03746

# ES
> ES(port.ret, p = 0.95, method="gaussian")
      port
ES -0.0332
> ES(port.ret, p = 0.99, method="gaussian")
      port
ES -0.04293
```

Bivariate Student's t Distribution

```

> library(MASS)      # needed for cov.trob
> library(mnormt)   # needed for dmt
> df = seq(2.1,5,.01) # grid of v values
> n = length(df)
> loglik_max = rep(0,n)
> for(i in 1:n)
+ {
+   fit = cov.trob(coredata(MSFT.GSPC.ret), nu=df[i])
+   loglik_max[i] = sum(log(dmt(coredata(MSFT.GSPC.ret),
+                               mean=fit$center,
+                               S=fit$cov,df=df[i])))
+ }
# find the maximum of the profile log-likelihood
> max.lik = max(loglik_max)

# find the value of v which maximizes the log-likelihood
> v.mle = df[which(loglik_max == max.lik)]

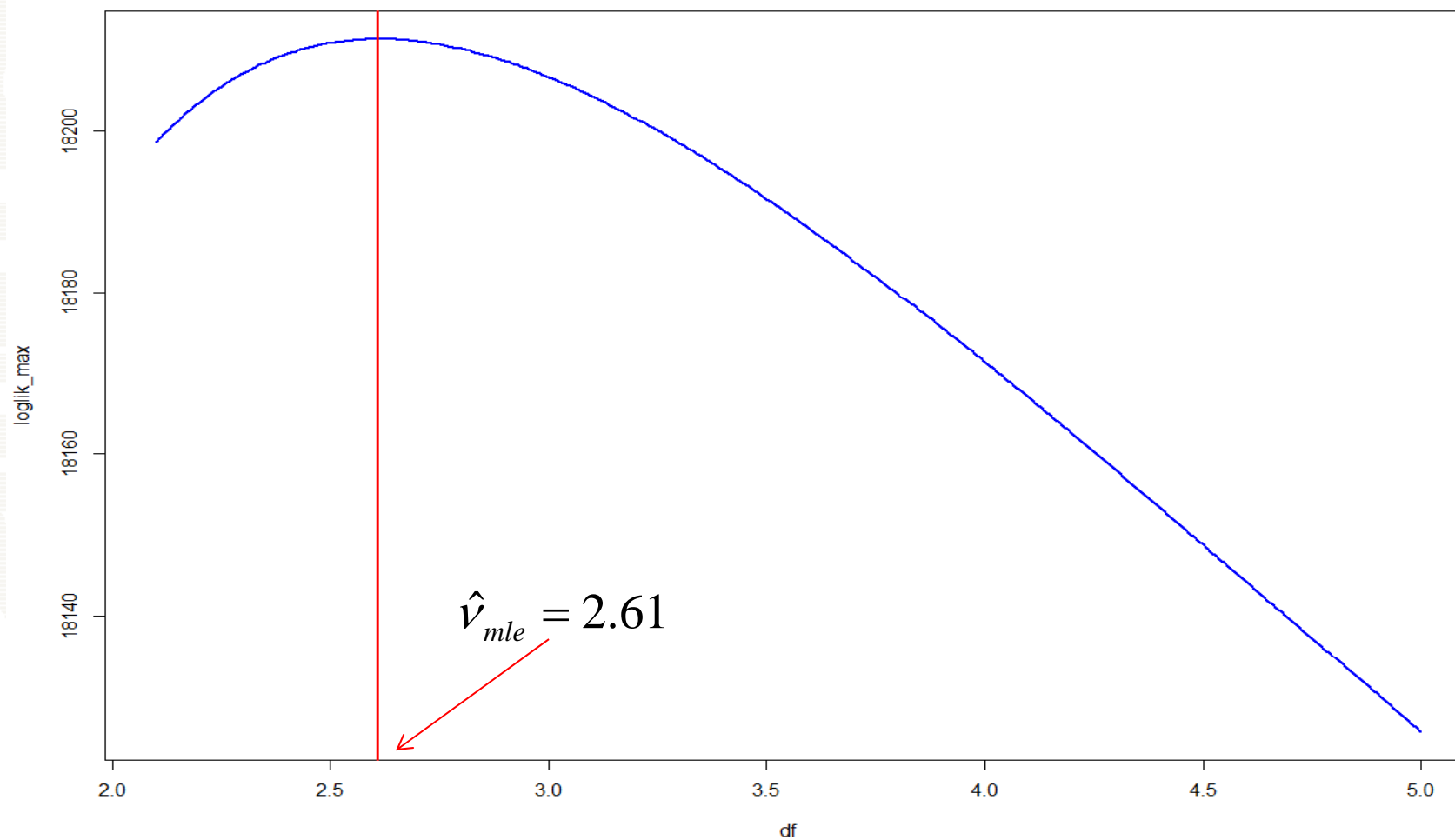
```

MLE of mu and cov given df

MLE of df given mu and cov

MLE of v in Bivariate Student's t

Profile Likelihood for Bivariate t



MLE of Bivariate Student t

```
# extract mle of mu and sigma given v.mle
> fit.mle = cov.trob(coredata(MSFT.GSPC.ret),nu=v.mle)
> mu.mle.t = fit.mle$center
> Sigma.mle.t = fit.mle$cov
> Cor.mle.t = cov2cor(Sigma.mle.t)

> mu.mle.t
MSFT.Adjusted  GSPC.Adjusted
      3.653e-05      3.364e-04

> Sigma.mle.t*(v.mle/(v.mle - 2))
              MSFT.Adjusted  GSPC.Adjusted
MSFT.Adjusted      0.0006388      0.0002998
GSPC.Adjusted      0.0002998      0.0002977

> Cor.mle.t
              MSFT.Adjusted  GSPC.Adjusted
MSFT.Adjusted      1.0000      0.6876
GSPC.Adjusted      0.6876      1.0000
```

Simulate Returns from Multivariate t

```
# initialize random number generator
> set.seed(123)

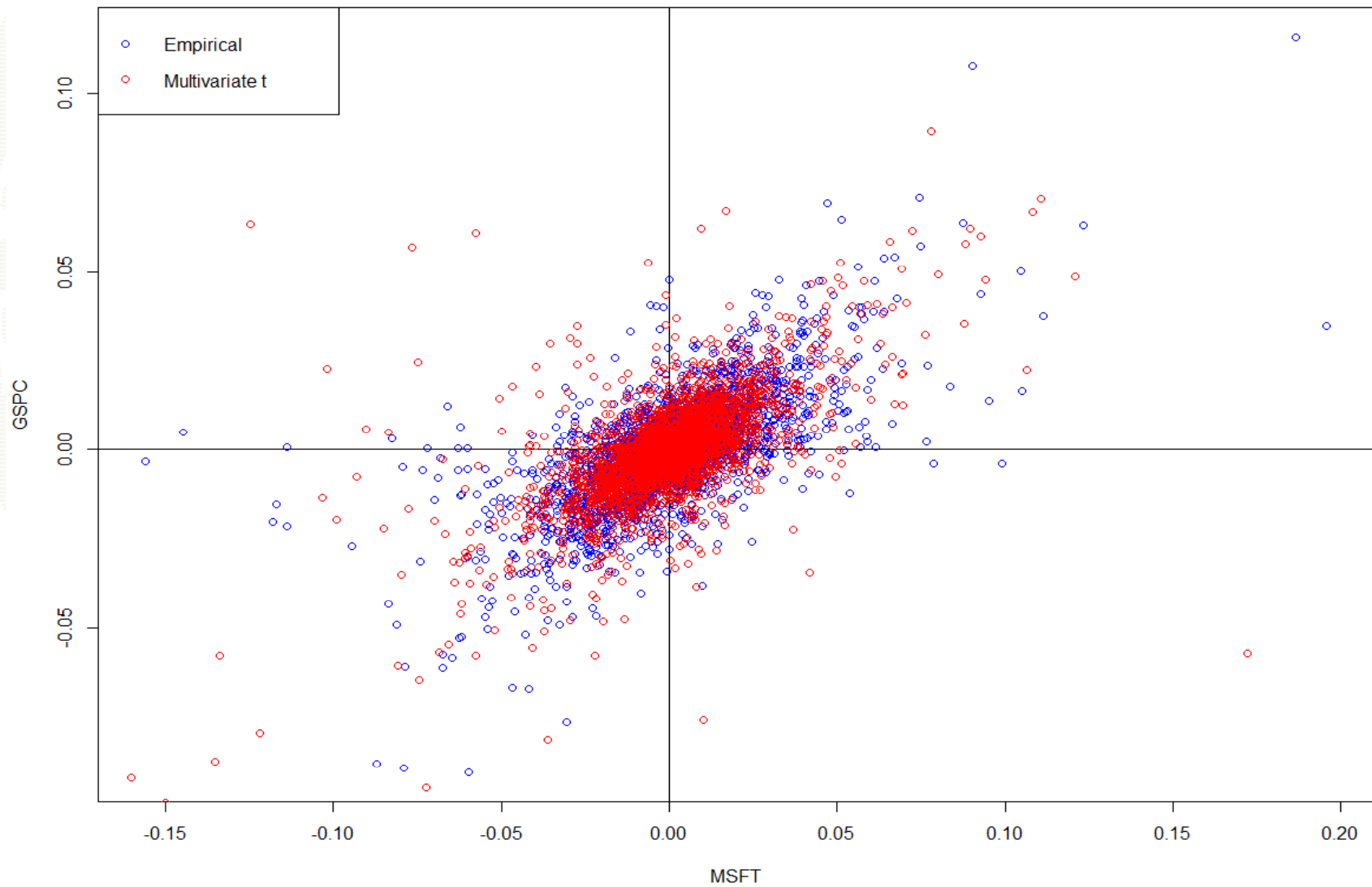
# Generate  $N(0, \text{Sigma})$  random variables
> Z = rmvnorm(n=n.obs, mean=c(0,0),
+           sigma=Sigma.mle.t*(v.mle/(v.mle - 2)))

# generate  $W \sim \text{chi-sq}(v.mle)$ 
> W = rchisq(n.obs,df=v.mle)

# simulate bivariate t
> sim.ret.t = mu.mle.t + sqrt(v.mle/W)*Z
> colnames(sim.ret.t) = c("MSFT","GSPC")
```


Simulate Bivariate t Returns

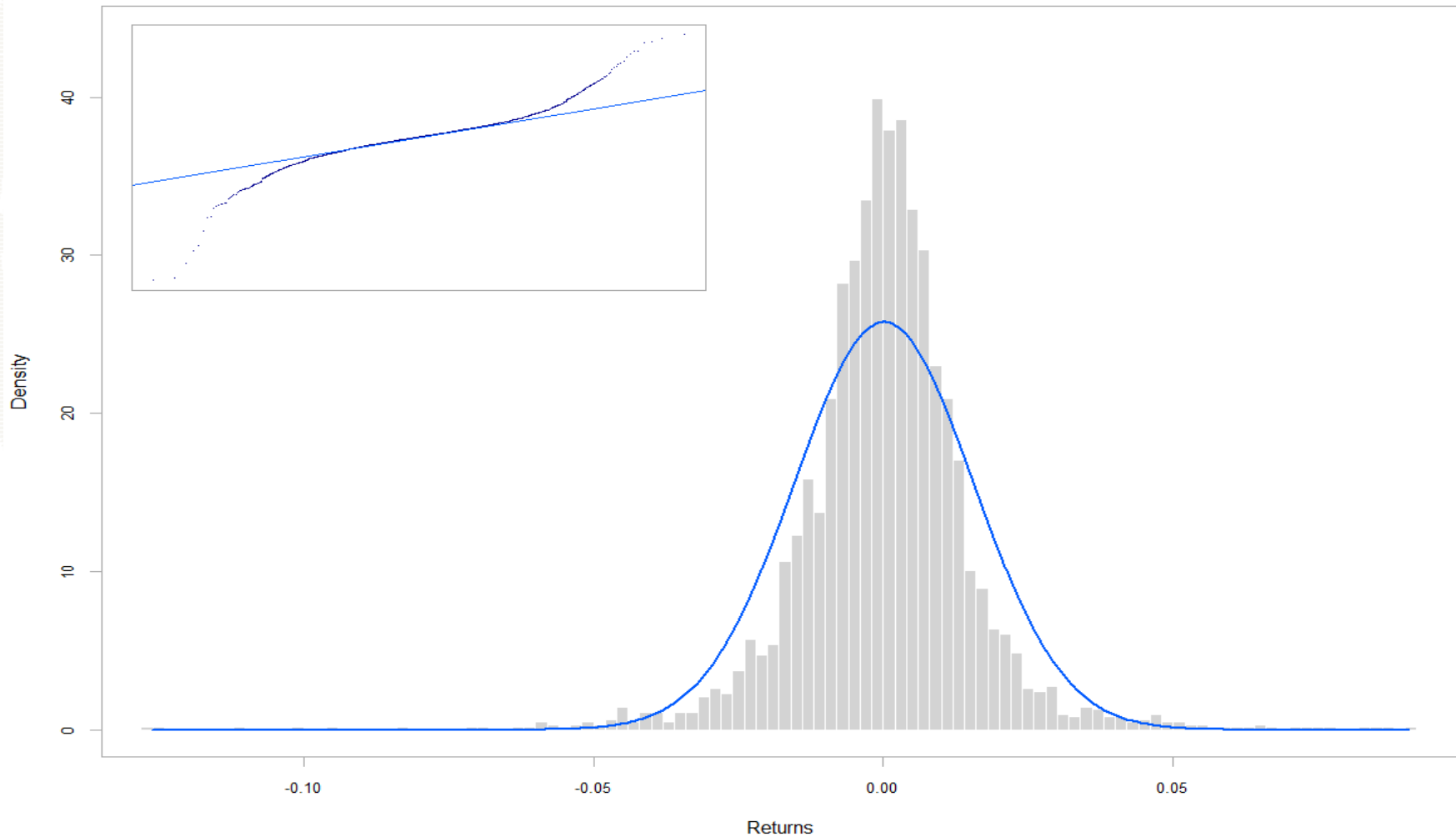
Empirical vs. Bivariate t



Simulated Portfolio Returns

```
> port.ret.sim.t = 0.5*sim.ret.t[, "MSFT"] + 0.5*sim.ret.t[, "GSPC"]
```

Equally Weighted Portfolio: Student's t



Risk Measures from Simulated Returns

```
# VaR
> VaR(port.ret.sim.t, p = 0.95, method="historical")
      [,1]
VaR -0.02283
> VaR(port.ret.sim.t, p = 0.99, method="historical")
      [,1]
VaR -0.04494

# ES
> ES(port.ret.sim.t, p = 0.95, method="historical")
      [,1]
ES -0.03699
> ES(port.ret.sim.t, p = 0.99, method="historical")
      [,1]
ES -0.06371
```

Risk Budgets from Simulated Returns

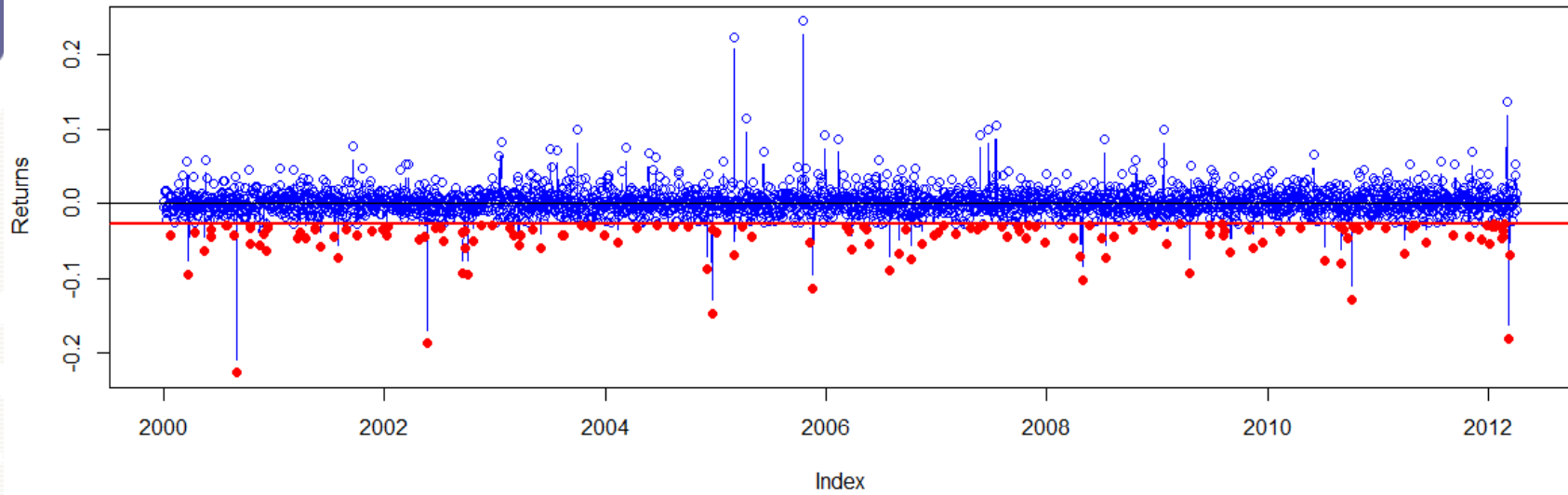
```
# Asset contributions to ES  
> ES(sim.ret.t, p=0.95, weights=c(0.5, 0.5),  
+ portfolio_method="component", method="historical")  
$`-r_exceed/c_exceed`
```

```
[1] 0.03709 ← Portfolio ES
```

```
$c_exceed  
[1] 154 ← # of returns less than VaR quantile
```

```
$realizedcontrib  
MSFT GSPC  
0.6207 0.3793 ← Percent contribution to ES
```

Portfolio Returns and 5% VaR Violations



Mean of MSFT when PORT ≤ 5% VaR

