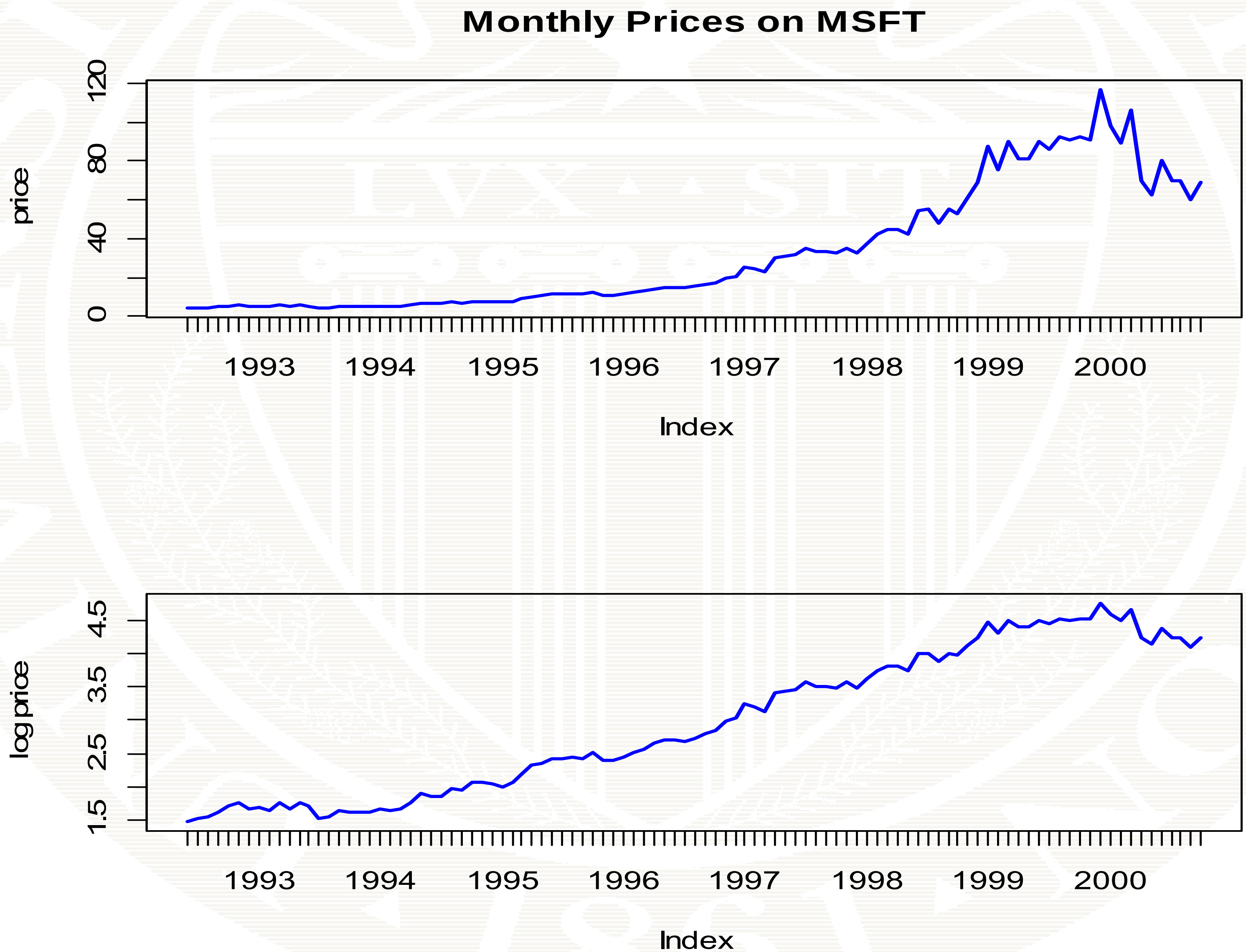
# Constant Expected Return Model

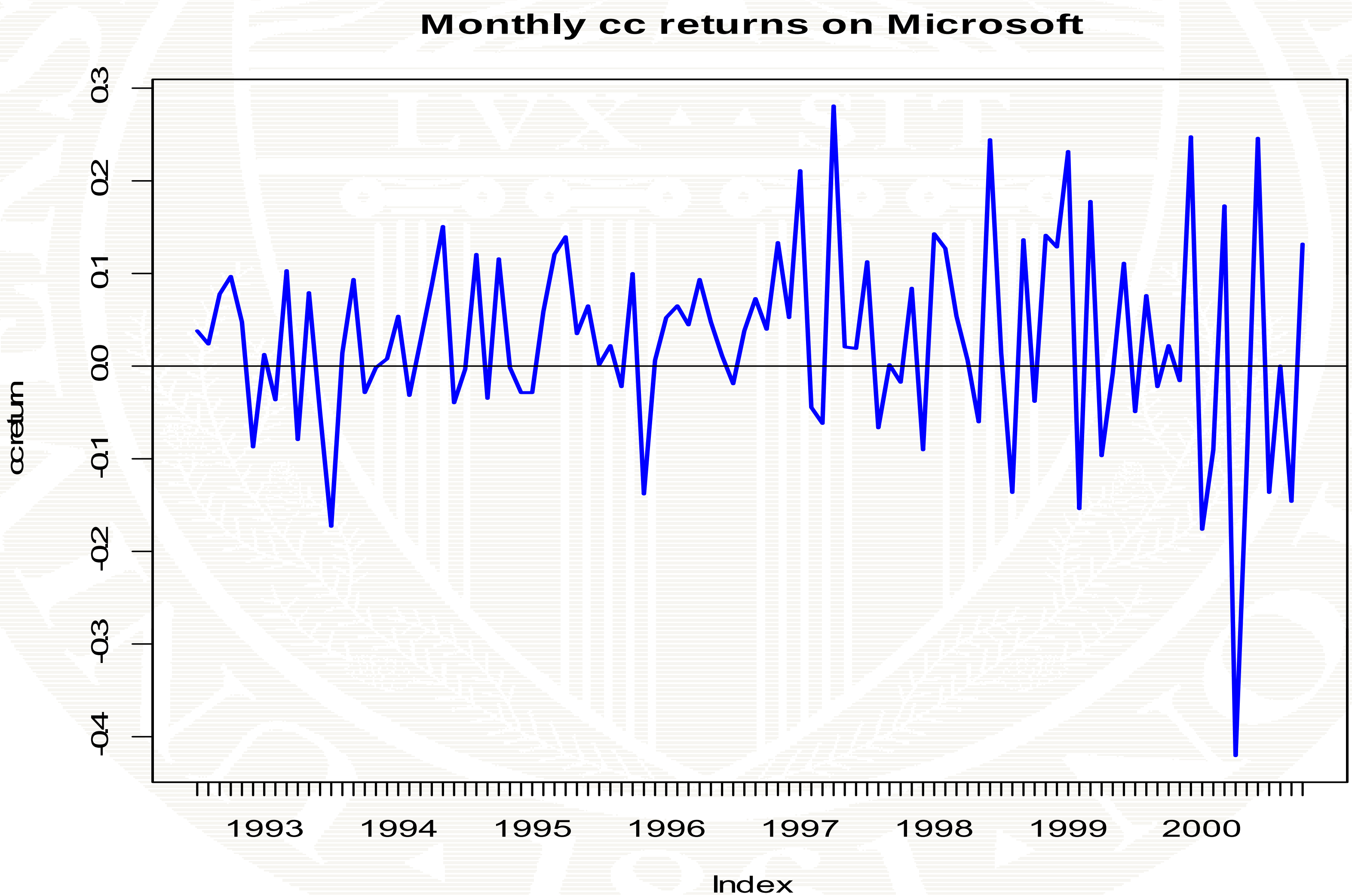
Econ 424
Eric Zivot
Fall 2014
Updated: October 23, 2014

# Monthly Closing Prices of Microsoft Stock

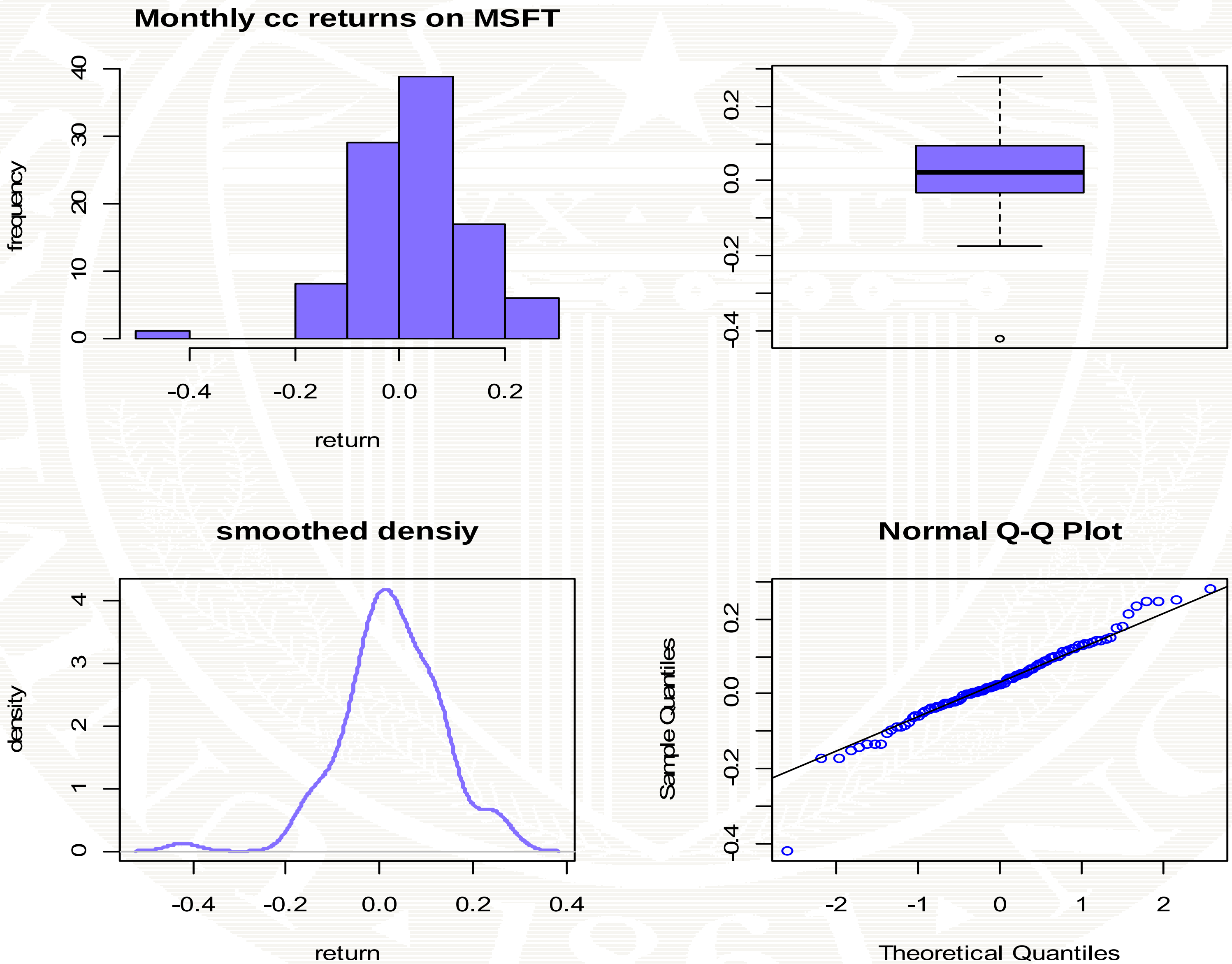


# Conjecture: Monthly CC Returns follow CER Model

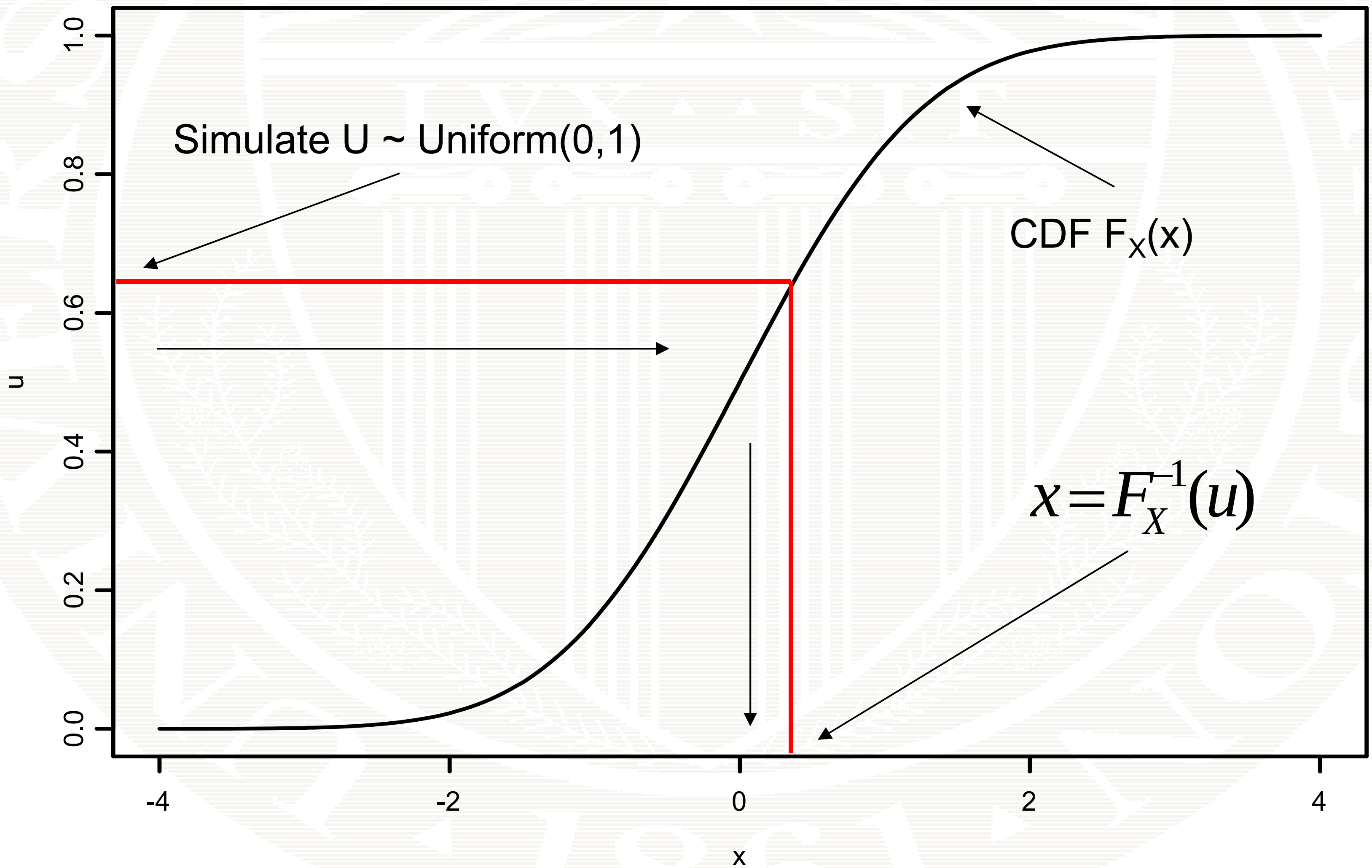
$r_t = \mu + \varepsilon_t$, $\varepsilon_t \sim$ iid $N(0,\sigma^2)$, t=July 1992,…,Oct 2000



# Distribution Summary for Monthly CC Returns on MSFT



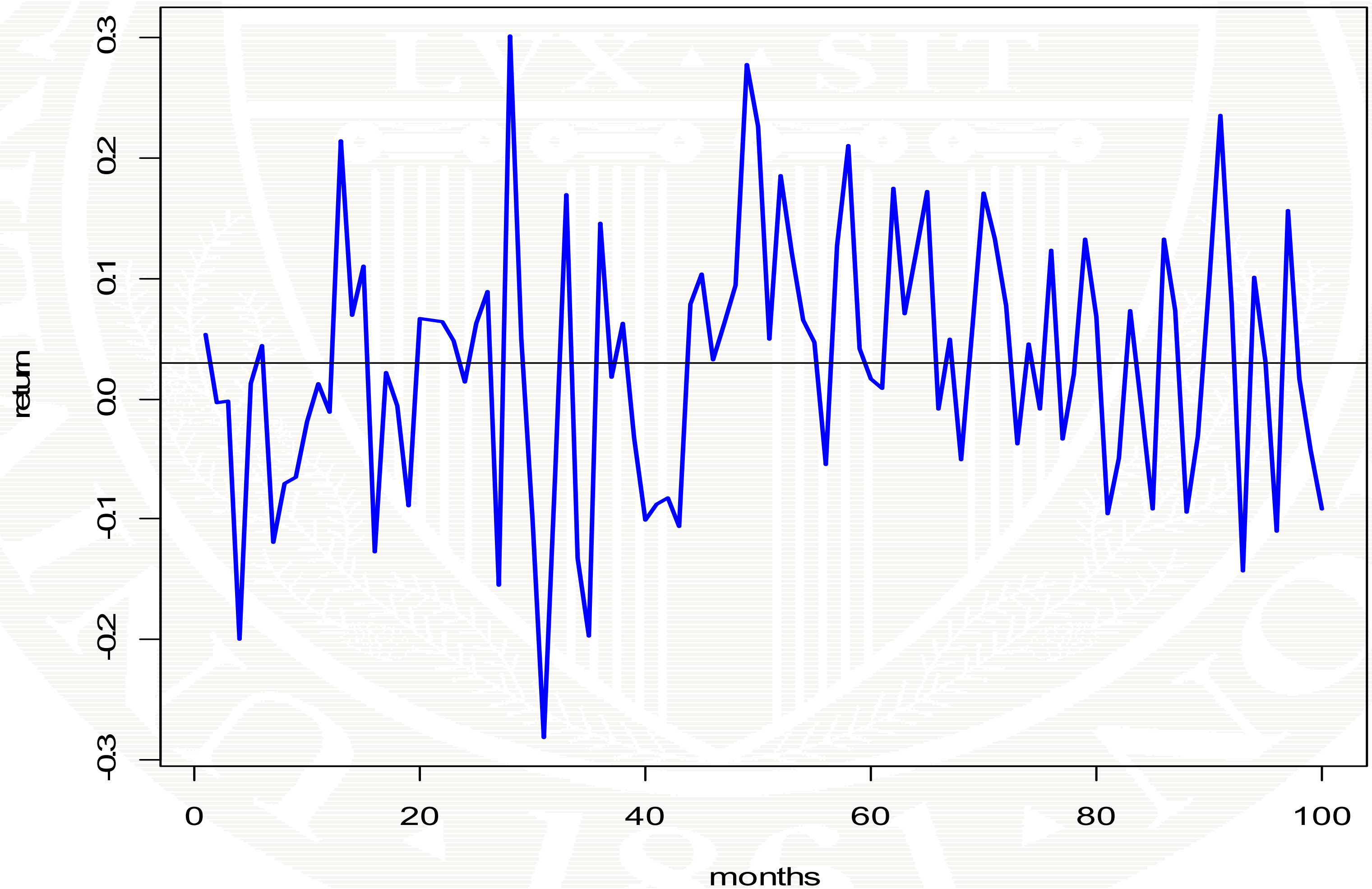
# Simulating Random Data for X with CDF $F_X$



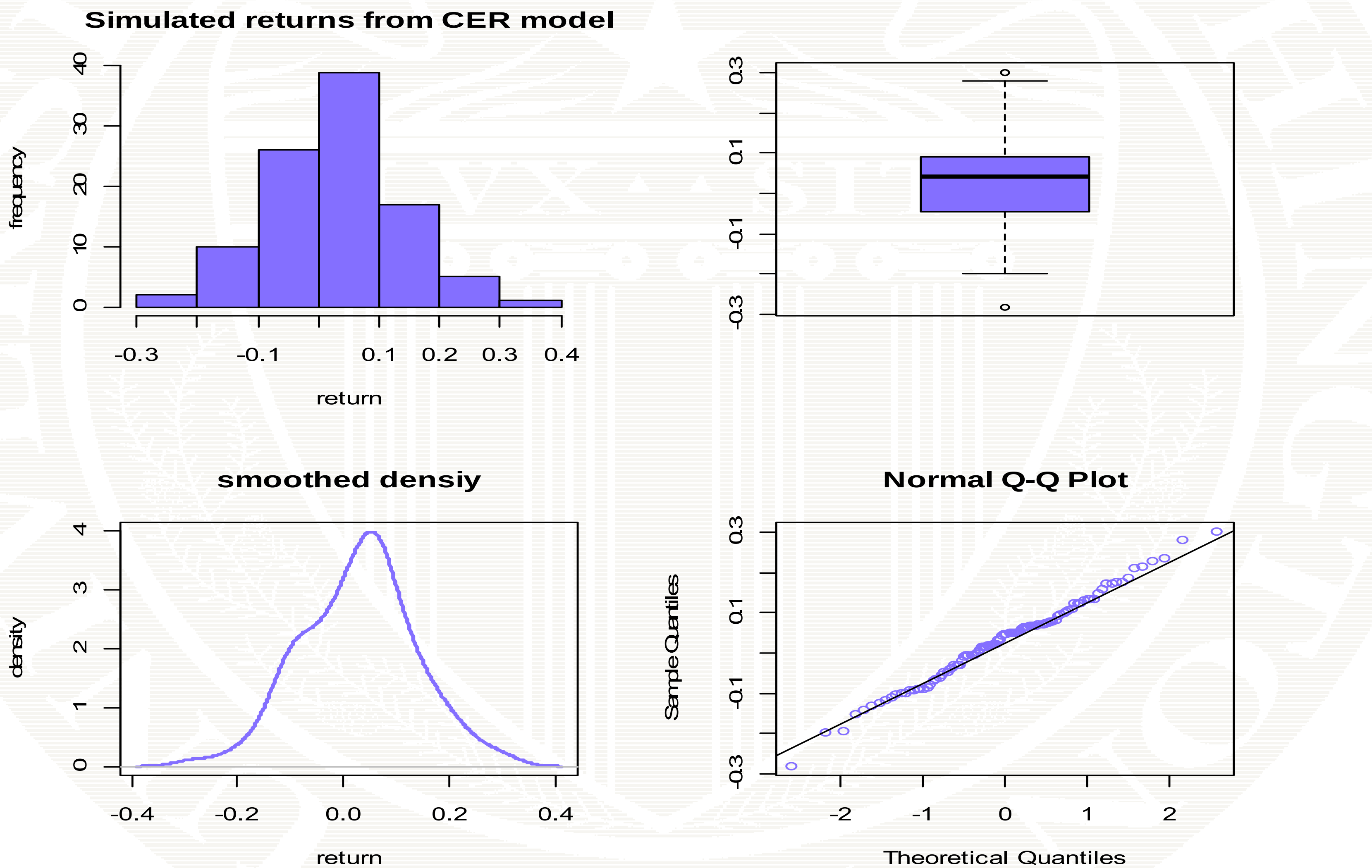
# R Code for MC Simulation of CER Model

```
# set model parameters
> mu = 0.03
> sd.e = 0.10
> nobs = 100

# generate random numbers for errors
> set.seed(111)
> sim.e = rnorm(nobs, mean=0, sd=sd.e)

# simulate cc returns
> sim.ret = mu + sim.e
```

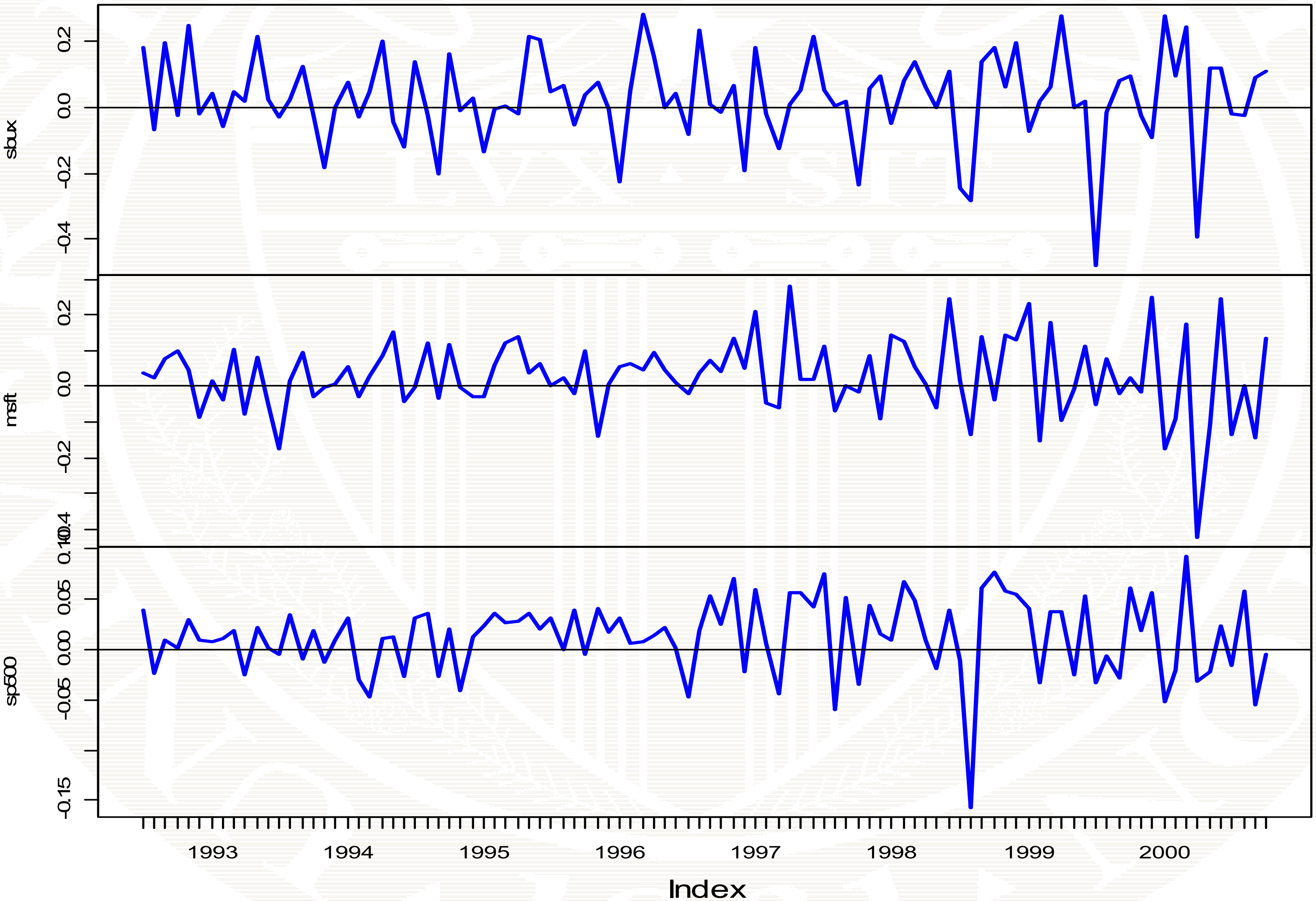
# Simulated Returns from CER Model with Same Mean and SD as Microsoft



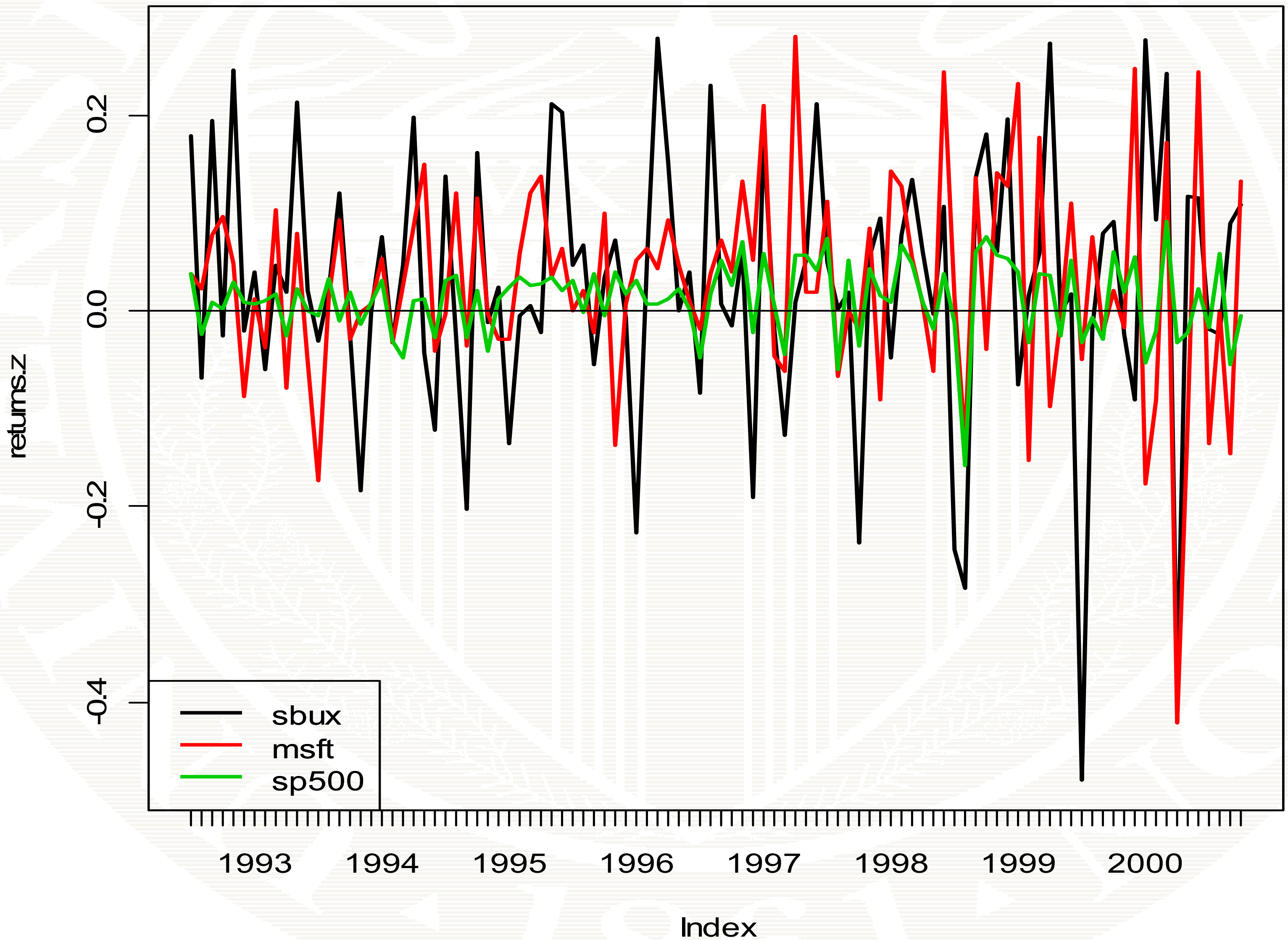
# Distribution Summary for Simulated Data from CER Model



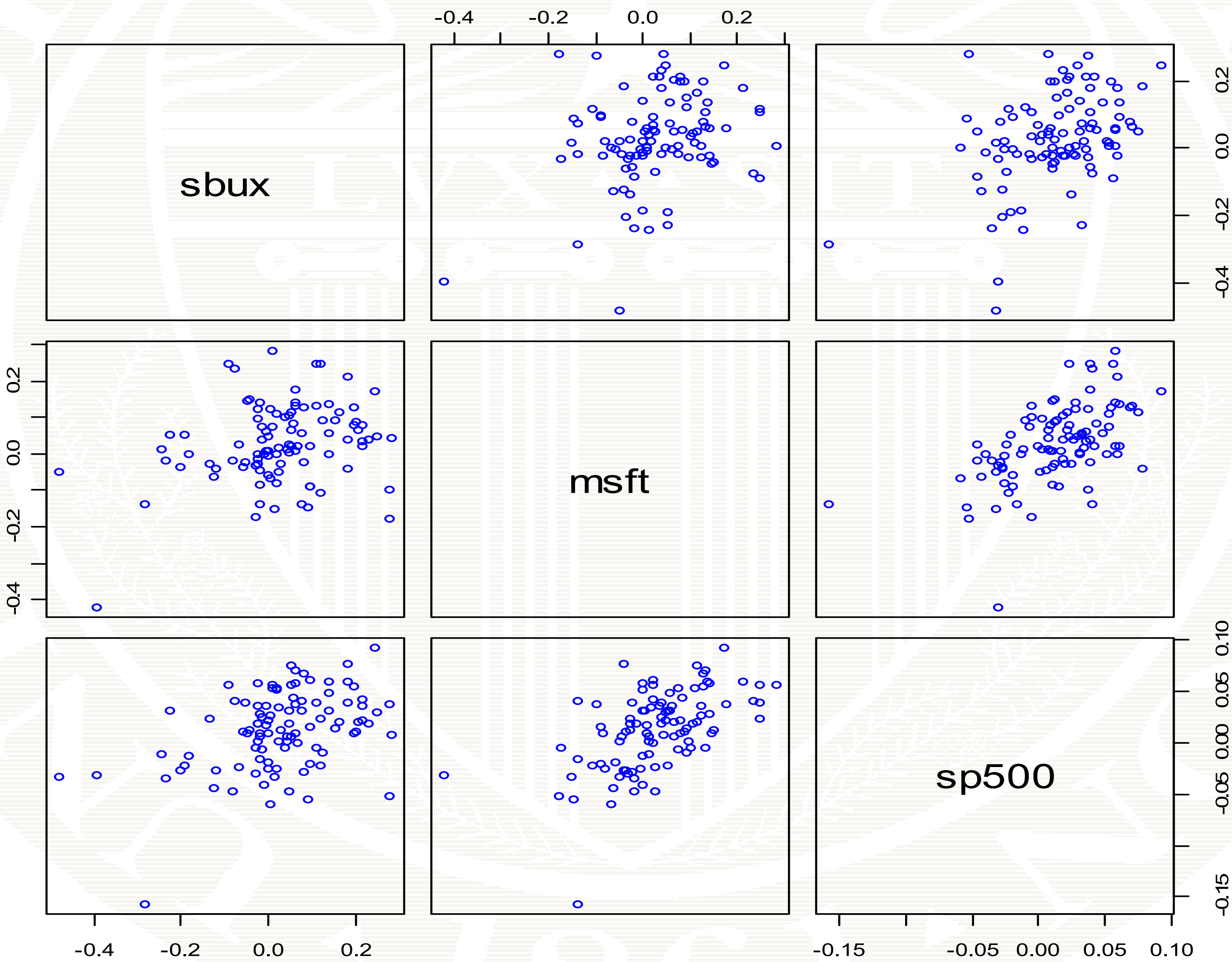
# Monthly Returns on MSFT, SBUX and S&P 500



# Monthly Returns on SBUX, MSFT and SP500



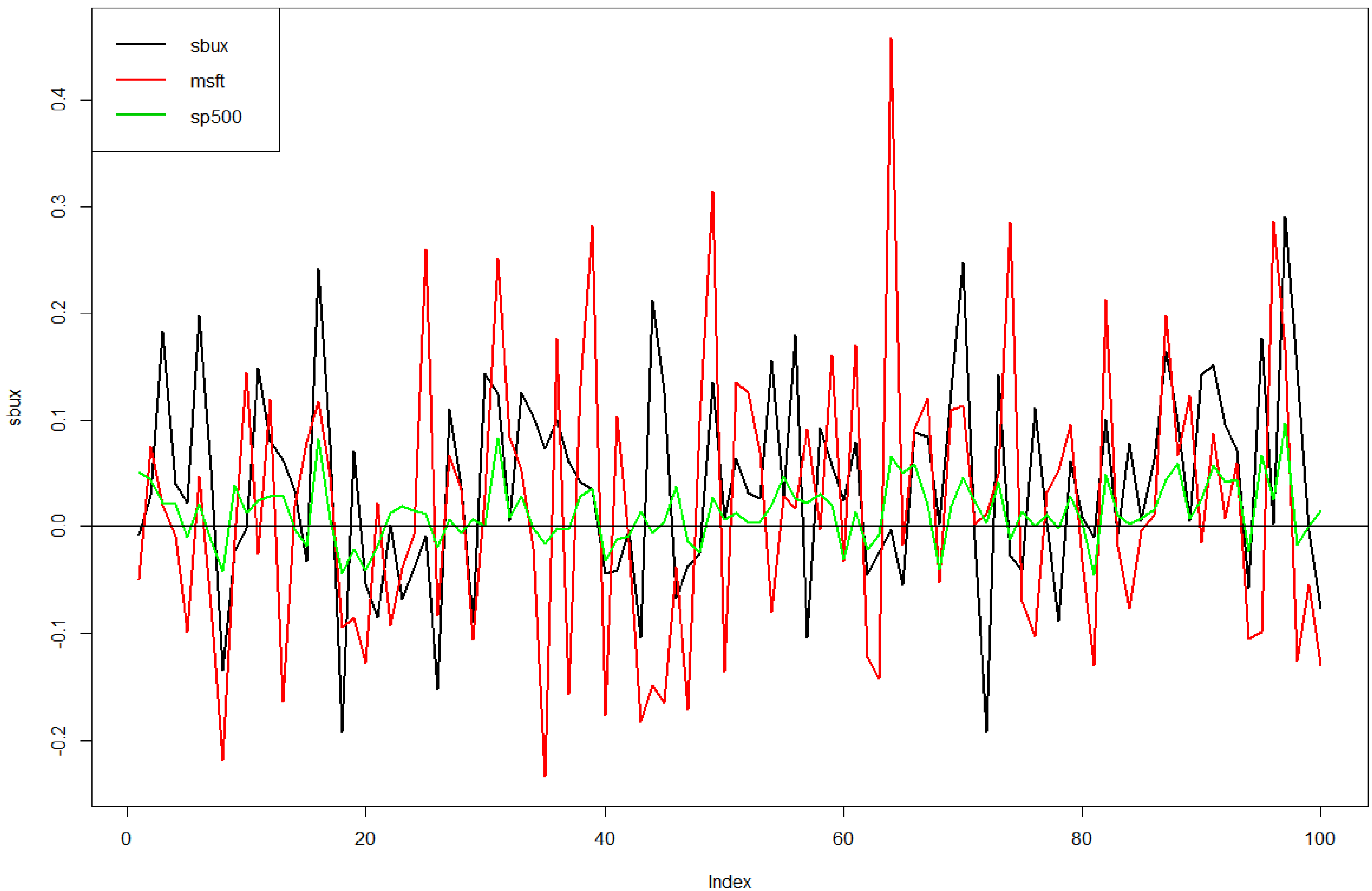
# Monthly Returns on SBUX, MSFT and SP500



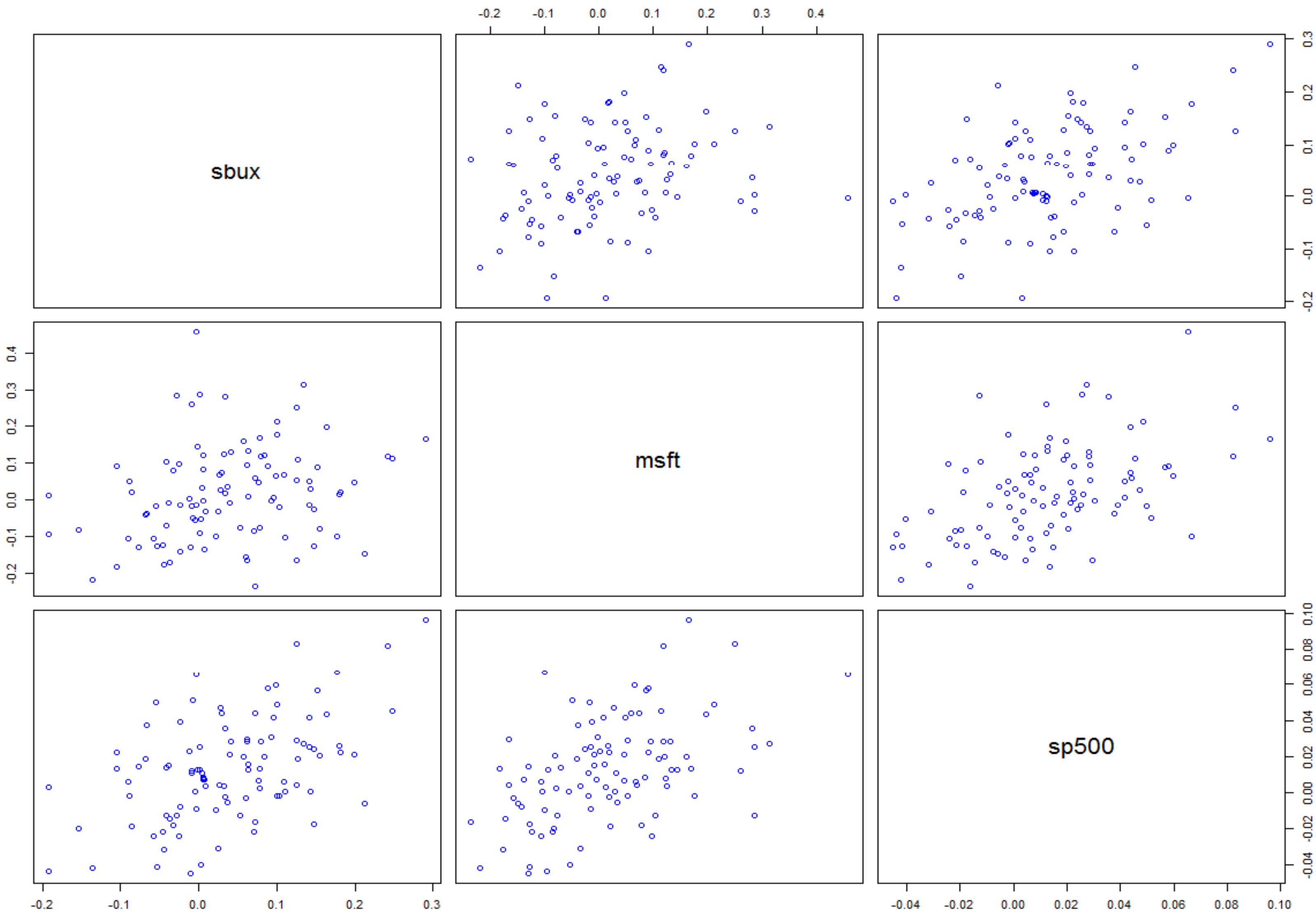
# R Code for MC Simulation for Multiple Assets

```
# multivariate simulation
> library("mvtnorm")
> mu = c(0.03,0.03,0.01)
> sig2.msft = 0.018
> sig2.sbux = 0.011
> sig2.sp500 = 0.001
> sig.msft.sbux = 0.004
> sig.msft.sp500 = 0.002
> sig.sbux.sp500 = 0.002
> Sigma = matrix(c(sig2.sbux, sig.msft.sbux, sig.sbux.sp500,
+                 sig.msft.sbux, sig2.msft, sig.msft.sp500,
+                 sig.sbux.sp500, sig.msft.sp500, sig2.sp500),
+                 nrow=3, ncol=3, byrow=TRUE)
> nobs = 100
> set.seed(123)
> returns.sim = rmvnorm(nobs, mean=mu, sigma=Sigma)
```

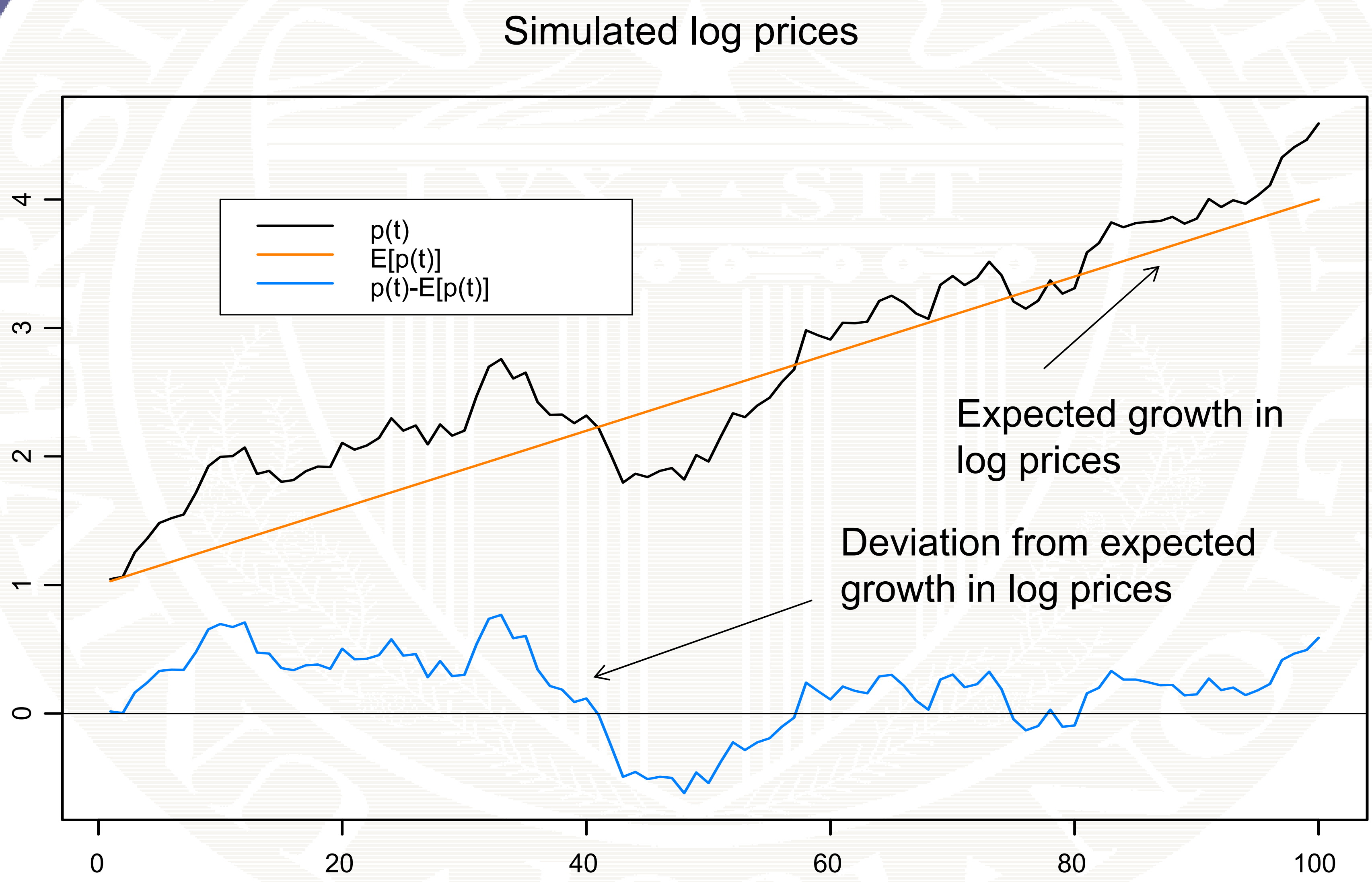
# Simulated Return Data



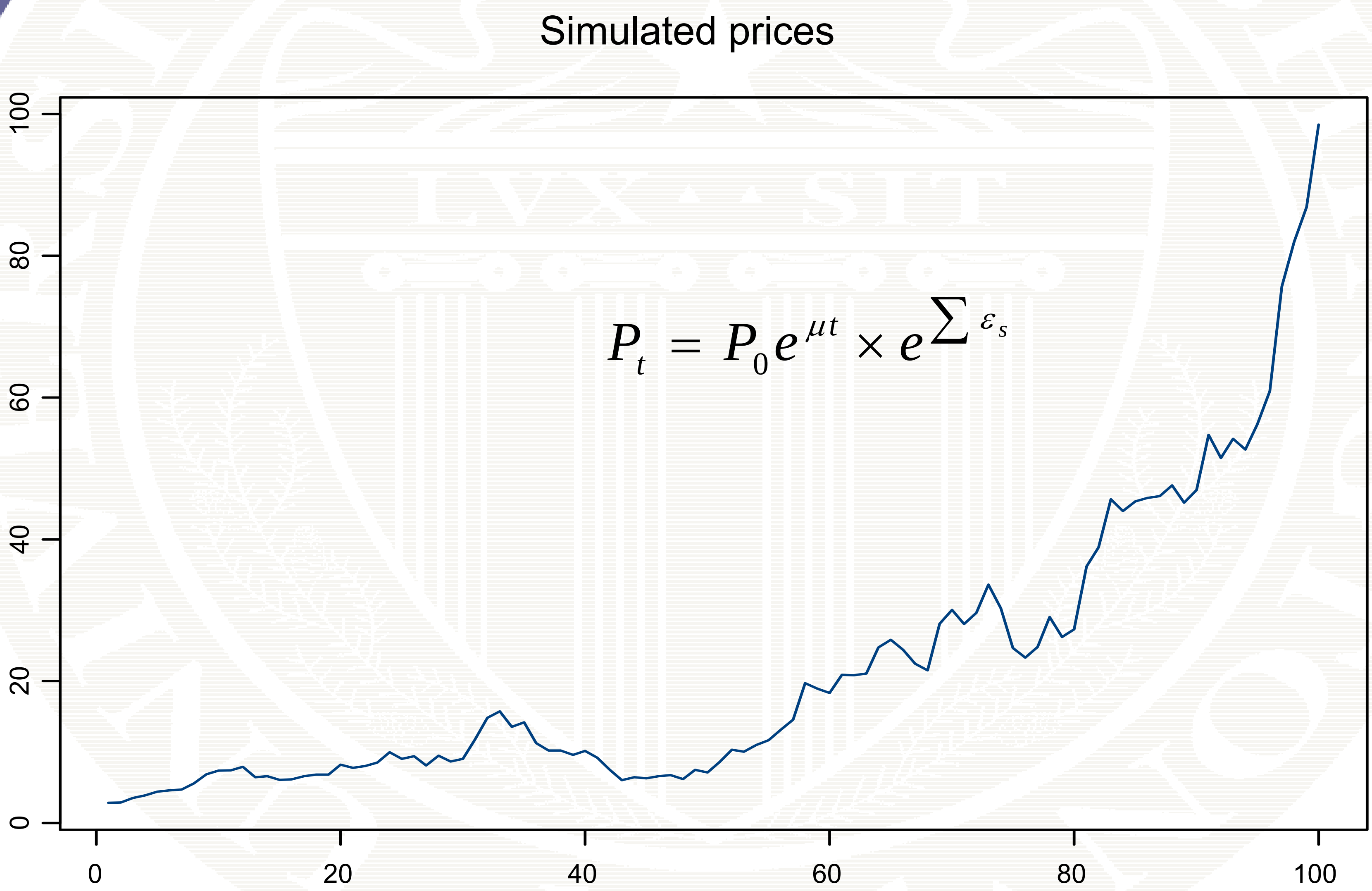
# Simulated Returns



# Simulated Data from Random Walk Model for Log Prices



# Simulated Prices from Random Walk Model




$$P_t = P_0 e^{\mu t} \times e^{\sum \varepsilon_s}$$

# CER Model Estimates

```
> muhat.vals = apply(returns.mat,2,mean)
> muhat.vals
    sbux    msft   sp500
 0.02777 0.02756 0.01253
```

$$\hat{\mu} = \frac{1}{T}\sum_{t=1}^{T} r_t$$

```
> sigma2hat.vals = apply(returns.mat,2,var)
> sigma2hat.vals
    sbux    msft    sp500
 0.01846 0.01141 0.001432
```

$$\hat{\sigma}^2 = \frac{1}{T-1}\sum_{t=1}^{T}(r_t - \hat{\mu})^2$$

```
> sigmahat.vals = apply(returns.mat,2,sd)
> sigmahat.vals
   sbux   msft   sp500
 0.1359 0.1068 0.03785
```

$$\hat{\sigma} = \sqrt{\hat{\sigma}^2}$$

# CER Model Estimates

$$\hat{\sigma}_{ij} = \frac{1}{T-1}\sum_{t=1}^{T}(r_{it} - \hat{\mu}_i)(r_{jt} - \hat{\mu}_j)$$

$$\hat{\rho}_{ij} = \frac{\hat{\sigma}_{ij}}{\hat{\sigma}_i \hat{\sigma}_j}$$

```
> cov.mat = var(returns.ts)
> cor.mat = cor(returns.ts)

> covhat.vals = cov.mat[lower.tri(cov.mat)]
> rhohat.vals = cor.mat[lower.tri(cor.mat)]
> names(covhat.vals) = names(rhohat.vals) =
+ c("sbux,msft","sbux,sp500","msft,sp500")

> covhat.vals
 sbux,msft sbux,sp500 msft,sp500
 0.00403   0.00215    0.00224

> rhohat.vals
 sbux,msft sbux,sp500 msft,sp500
   0.2777  0.4197     0.5551
```

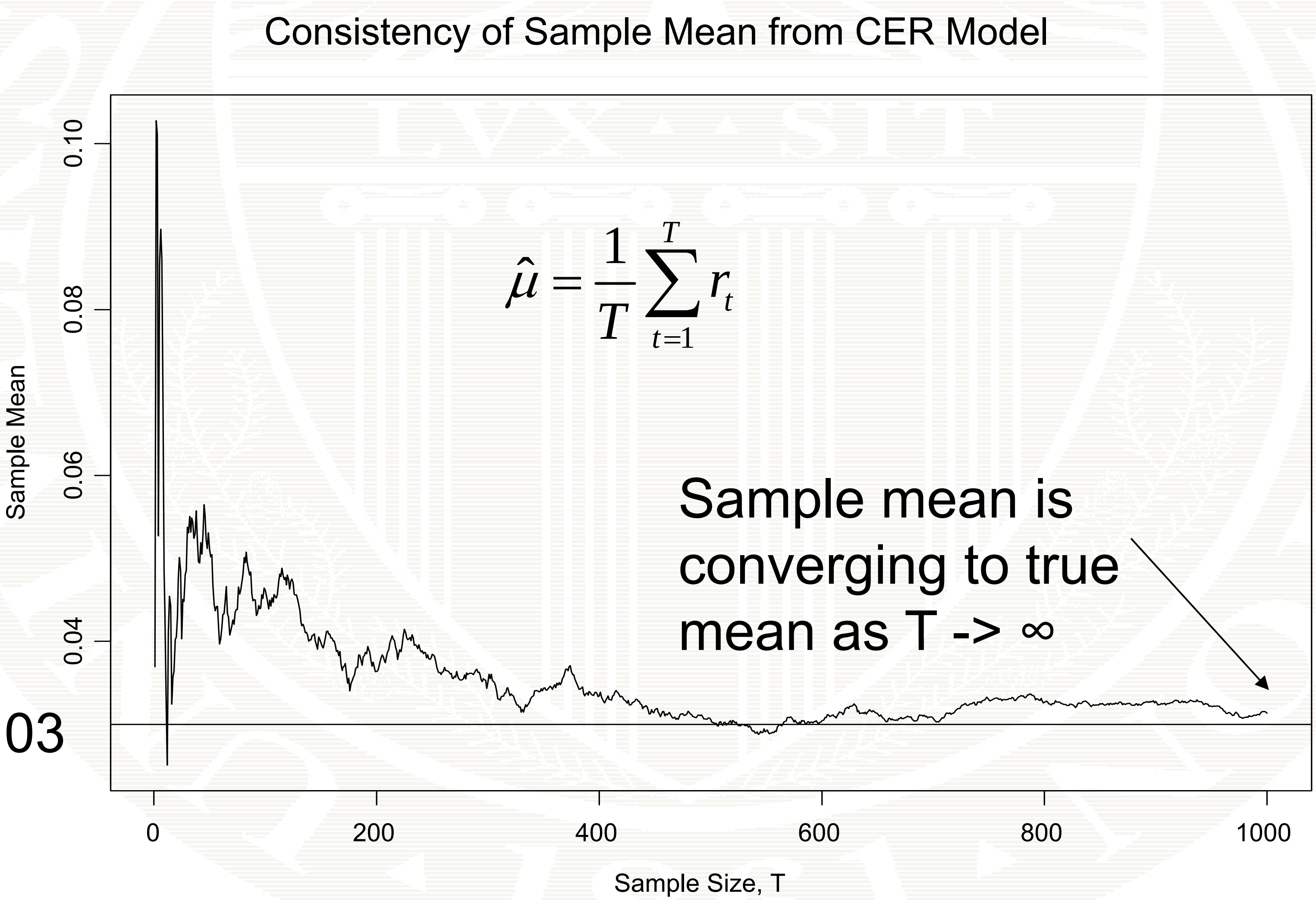
# Estimated Standard Errors

```
> se.muhat = sigmahat.vals/sqrt(nobs)
> rbind(muhat.vals,se.muhat)
               sbux    msft    sp500
muhat.vals     0.0277  0.0275  0.01253
  se.muhat     0.0135  0.0106  0.00378

> se.sigma2hat = sigma2hat.vals/sqrt(nobs/2)
> rbind(sigma2hat.vals,se.sigma2hat)
                  sbux     msft     sp500
sigma2hat.vals    0.01845  0.01141  0.00143
  se.sigma2hat    0.00261  0.00161  0.00020

> se.sigmahat = sigmahat.vals/sqrt(2*nobs)
> rbind(sigmahat.vals,se.sigmahat)
                 sbux    msft    sp500
sigmahat.vals    0.1358  0.1068  0.0378
  se.sigmahat    0.0096  0.0075  0.0026
```

# Estimated Standard Errors

```
> se.rhohat = (1-rhohat.vals^2)/sqrt(nobs)
> rbind(rhohat.vals,se.rhohat)
            sbux,msft sbux,sp500 msft,sp500
rhohat.vals 0.2777    0.4197     0.5551
  se.rhohat 0.0922    0.0823     0.0691
```

# Sample means computed with increasing sample sizes




$\mu = 0.03$

# 95% Confidence Intervals For $\mu$

```
> mu.lower = muhat.vals - 2*se.muhat
> mu.upper = muhat.vals + 2*se.muhat
> mu.width = mu.upper - mu.lower

> cbind(mu.lower,mu.upper,mu.width)
      mu.lower mu.upper mu.width
 sbux   0.0006   0.0549   0.0543
 msft   0.0061   0.0489   0.0427
sp500   0.0049   0.0201   0.0151
```

Wide 95% confidence intervals for the mean => imprecise estimate. Note: width of CI is large relative to size of estimate for sbux and msft

# 95% Confidence Intervals for $\sigma$

```
> sigma.lower = sigmahat.vals - 2*se.sigmahat
> sigma.upper = sigmahat.vals + 2*se.sigmahat
> sigma.width = sigma.upper - sigma.lower
```

```
> cbind(sigma.lower,sigma.upper,sigma.width)
      sigma.lower sigma.upper sigma.width
 sbux 0.1166      0.1550      0.0384
 msft 0.0917      0.1219      0.0302
sp500 0.0324      0.0431      0.0107
```

Narrow 95% confidence intervals for the sd => precise estimate. Note: width of CI is small relative to value of estimate

# 95% Confidence Intervals for $\rho$

```
> rho.lower = rhohat.vals - 2*se.rhohat
> rho.upper = rhohat.vals + 2*se.rhohat
> rho.width = rho.upper - rho.lower

> cbind(rho.lower,rho.upper,rho.width)
            rho.lower rho.upper rho.width
 sbux,msft   0.0931    0.4622    0.3691
sbux,sp500   0.2549    0.5845    0.3295
msft,sp500   0.4167    0.6934    0.2767
```

95% confidence interval for rho is moderately large => somewhat imprecise estimate for rho.

# Stylized Facts for the Estimation of CER Model Parameters

- The expected return is not estimated very precisely
  - Large standard errors relative to size of mean estimates
- Standard deviations and correlations are estimated more precisely than the expected return

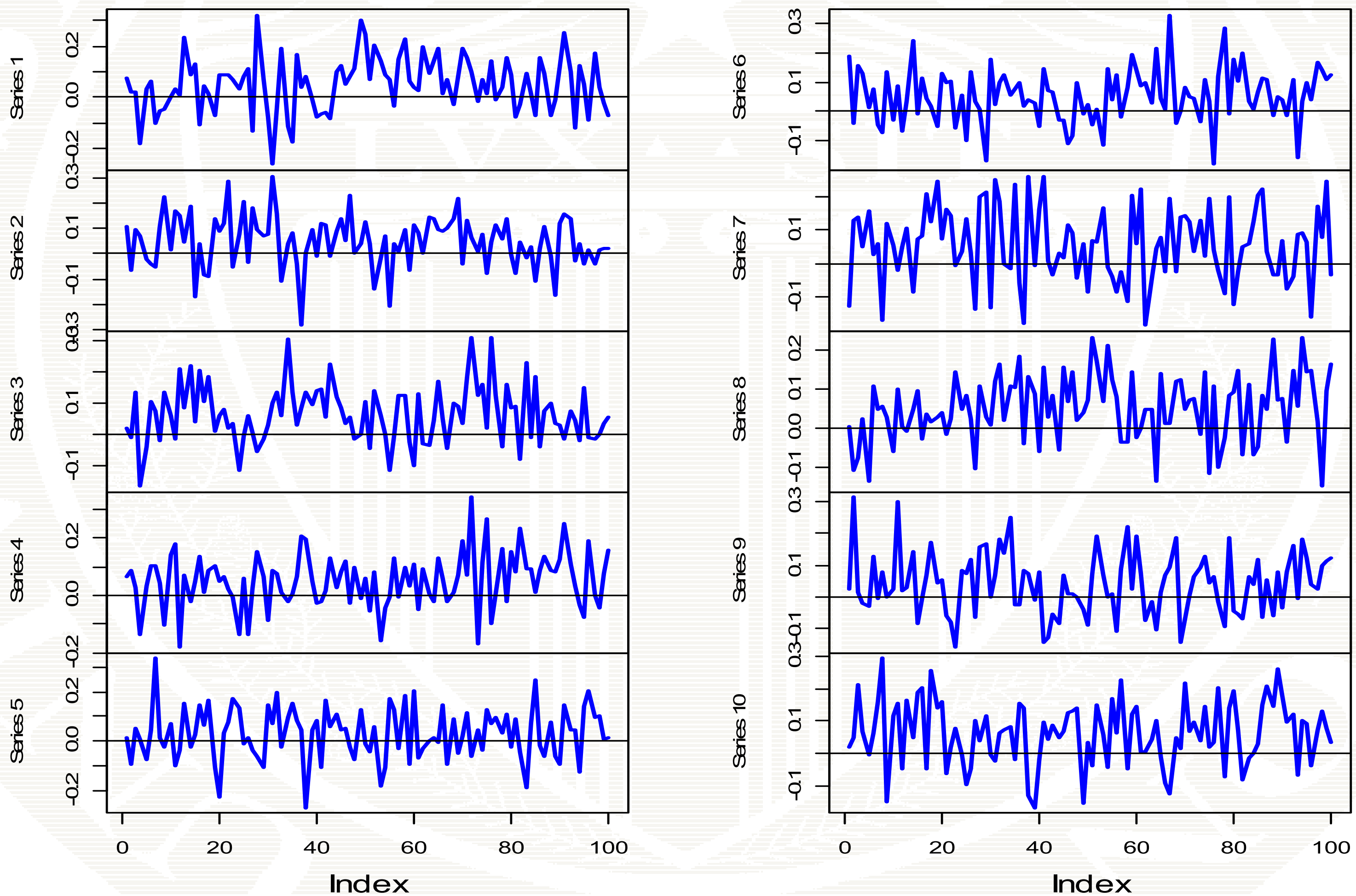
# Monte Carlo Simulation Loop

$r_{it} = 0.05 + \varepsilon_{it} \quad t = 1, .., 100$

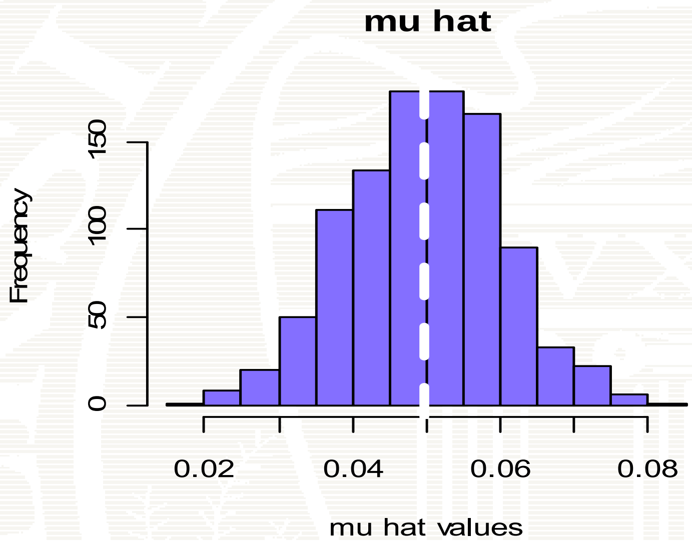
$\varepsilon_{it} \sim iid\ N(0, (0.10)^2)$

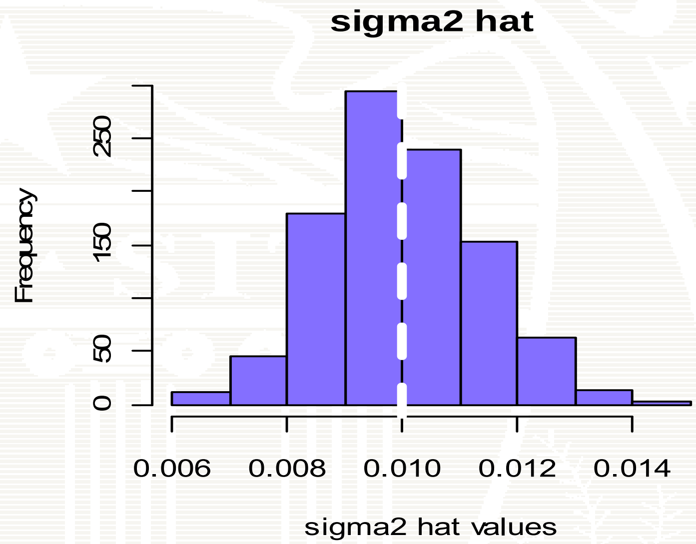
```
> mu = 0.05
> sd = 0.10
> n.obs = 100
> n.sim = 1000
> set.seed(111)
> sim.means = rep(0,n.sim)
> sim.vars = rep(0,n.sim)
> sim.sds = rep(0,n.sim)
> for (sim in 1:n.sim) {
     sim.ret = rnorm(n.obs,mean=mu,sd=sd)
     sim.means[sim] = mean(sim.ret)
     sim.vars[sim] = var(sim.ret)
     sim.sds[sim] = sqrt(sim.vars[sim])
  }
```

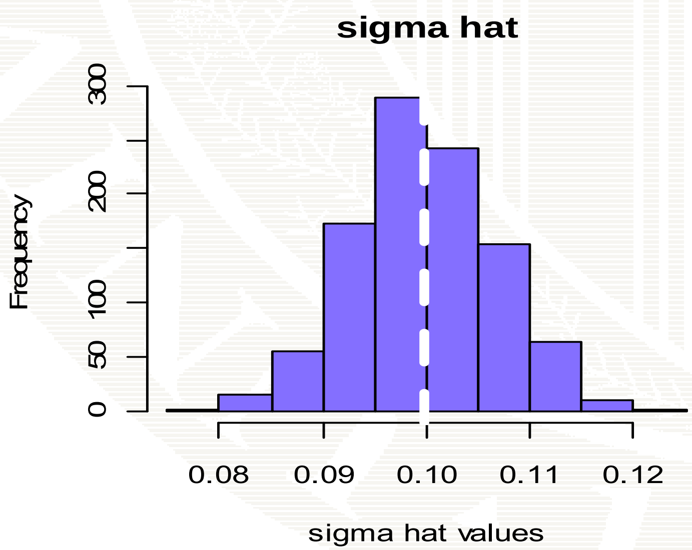
# 10 simulated samples from CER model



# Histograms of 1000 Monte Carlo Estimates




True values:

$E[R] = \mu = 0.05$

$SD(R) = \sigma = 0.10$

$Var(R) = \sigma^2 = 0.01$

# Monte Carlo Evaluation of Bias

```
> mean(sim.means)         # true mean = 0.05
[1] 0.04969
> mean(sim.means) – mu   # estimate of bias
[1] -0.0003105

> mean(sim.vars)          # true variance = 0.01
[1] 0.00999
> mean(sim.vars) - sd^2 # estimate of bias
[1] -9.865e-06

> mean(sim.sds)           # true SD = 0.10
[1] 0.09972
> mean(sim.sds) – sd     # estimate of bias
[1] -0.0002782
```

# Monte Carlo Evaluation of Estimated Standard Error

```
> sd(sim.means)    # SD of mu estimates across 1000
[1] 0.01041        # Monte Carlo experiments
> sd/sqrt(nobs)
[1] 0.01           # true SE estimate from formula

> sd(sim.vars)     # SD of sigma^2 estimates across 1000
[1] 0.001352       # Monte Carlo experiments
> sd^2/sqrt(nobs/2)
[1] 0.001414       # approx SE estimate from formula

> sd(sim.sds)      # SD of sigma estimates across 1000
[1] 0.006764       # Monte Carlo experiments
> sd/sqrt(2*nobs)
[1] 0.007071       # approx SE estimate from formula
```

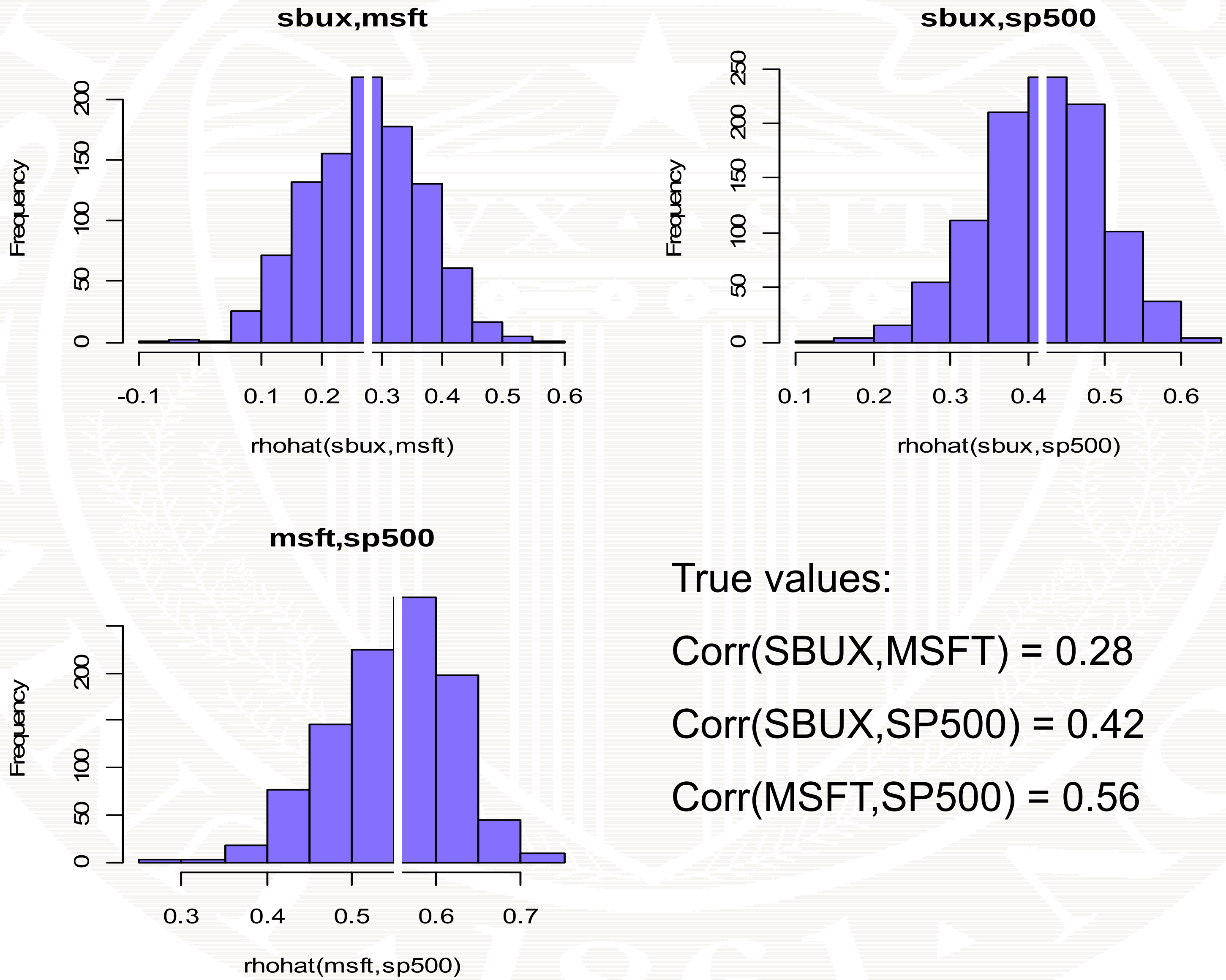
# Monte Carlo Evaluation of 95% Confidence Interval Coverage

```
> u = 0.05
> sd = 0.10
> n.sim = 1000
> set.seed(111)
> mu.lower = rep(0,n.sim)     # initialize vectors
> mu.upper = rep(0,n.sim)
> for (sim in 1:n.sim) {
      sim.ret = rnorm(n.obs,mean=mu,sd=sd)
      mu.hat = mean(sim.ret)
      se.muhat = sd(sim.ret)/sqrt(n.obs)
      mu.lower[sim] = mu.hat - 2*se.muhat
      mu.upper[sim] = mu.hat + 2*se.muhat
}
> in.interval = (mu >= mu.lower) & (mu <= mu.upper)
> sum(in.interval)/n.sim

>  0.934 # coverage probability of 95% CI for mu
```

# Monte Carlo Simulation Loop to Evaluate Distribution of correlation estimate

```
# generate 1000 samples from CER and compute correlations
# use estimated parameters as true parameters for MC
> n.obs = 100
> n.sim = 1000
> set.seed(111)
> sim.corrs = matrix(0,n.sim,3)     # initialize vectors
> colnames(sim.corrs) = c("sbux,msft", "sbux,sp500",
                          "msft,sp500")

> for (sim in 1:n.sim) {
      sim.ret = rmvnorm(n.obs, mean=muhat.vals,
                        cov=cov.mat)
      cor.mat = cor(sim.ret)
      sim.corrs[sim,] = cor.mat[lower.tri(cor.mat)]
  }
```

# Histograms of 1000 Monte Carlo Estimates of Correlation




True values:

Corr(SBUX,MSFT) = 0.28

Corr(SBUX,SP500) = 0.42

Corr(MSFT,SP500) = 0.56

# Monte Carlo Evaluation

```
# true correlation values
 sbux,msft sbux,sp500 msft,sp500
    0.2777     0.4198     0.5551
# Averages across 1000 Monte Carlos
> apply(sim.corrs,2,mean)
 sbux,msft sbux,sp500 msft,sp500
     0.277     0.4176     0.5505
# Monte Carlo Standard Deviations
> apply(sim.corrs,2,sd)
 sbux,msft sbux,sp500 msft,sp500
   0.09606    0.08148    0.07244
# Analytic SE values for rhohat
 sbux,msft sbux,sp500 msft,sp500
   0.09229    0.08238    0.06919
```

# Estimating VaR in CER Model

```
# estimate quantiles from CER model
> qhat.05 = muhat.vals + sigmahat.vals*qnorm(0.05)
> qhat.05
     sbux     msft     sp500
 -0.19571 -0.14815 -0.049717

# estimate 5% VaR
> W0 = 100000
> VaRhat.05 = (exp(qhat.05)-1)*W0
> VaRhat.05
   sbux   msft   sp500
 -17775 -13769 -4850.1
```