Macro & Markets In R

James Picerno | CapitalSpectator.com

22 October 2013

New Jersey

Hilton Woodbridge, Iselin, NJ

- Bootstrapping
  S&P 500: estimating return distribution based on existing sample: 1957-2013

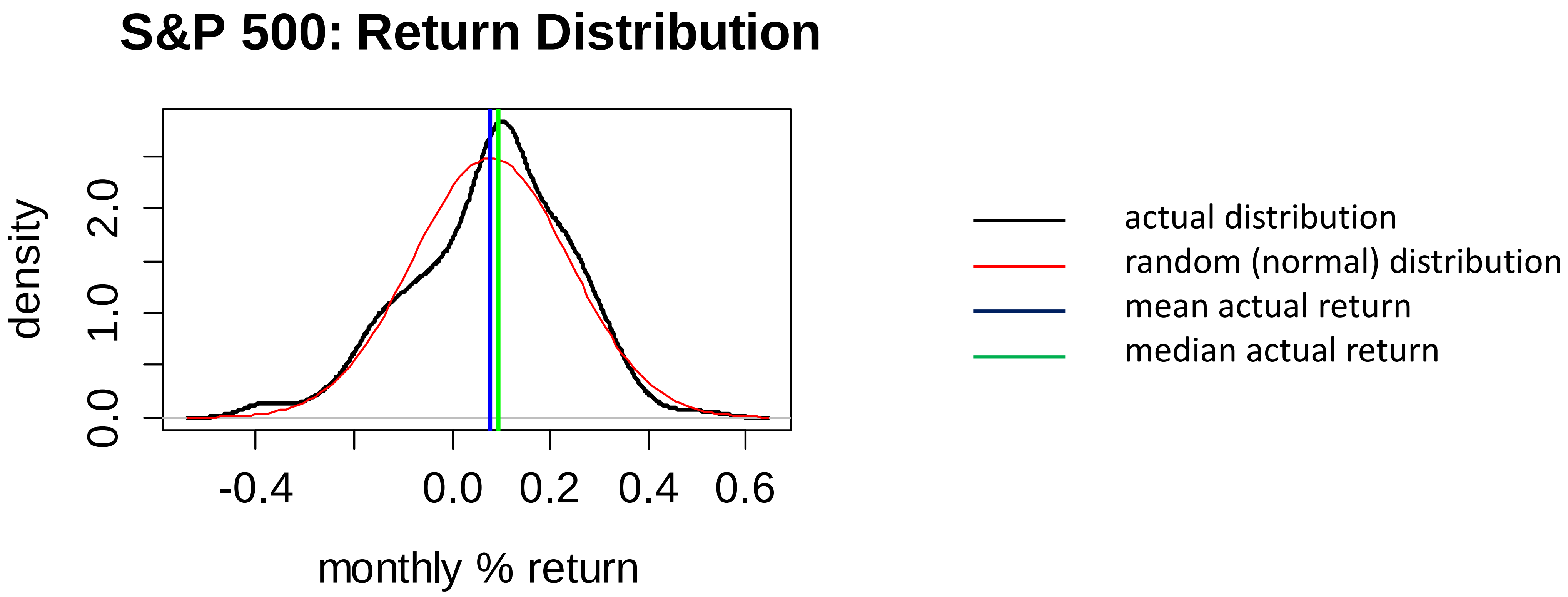
- Monte Carlo Simulation
  Simulating returns for analyzing five-year-ahead windows

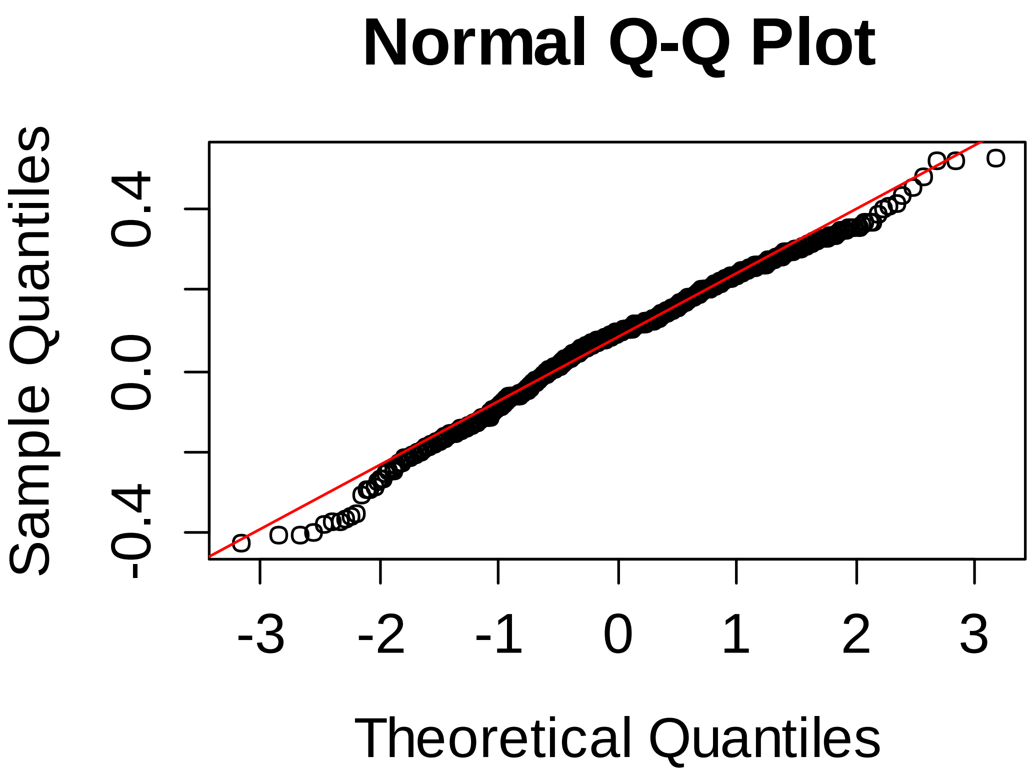
- Forecasting Economic Data With ARIMA
  Using the forecast package to project economic indicators,
  and combining predictions with historical data via a customized function

Bootstrapping

```
# Load packages
library(zoo)
library(PerformanceAnalytics)
library(tseries)
library(quantmod)

# Download & adjust S&P 500 index (daily) via quantmod package
fred.tickers <-c("SP500")
getSymbols(fred.tickers,src="FRED")

# Remove NAs
SP500 <-na.omit(SP500)

# Create monthly data & monthly return series (in numeric format)
SP500.m <-aggregate(SP500, (as.yearmon(time(SP500))), mean)
SP500.chg <-na.omit(as.numeric(ROC(SP500.m,12,"discrete")))

# Generate data set of random returns
x=rnorm(1:10000) # (for use in dnorm()) #10,000 random samples
```

```
# Graph of actual S&P 500 monthly % return distribution (black line)
# vs. random/normally distributed % return series (red line)

plot(density(SP500.chg), lwd=2, col="black",
    main="S&P 500: Return Distribution",
    xlab="monthly % return",
    ylab="density")
curve(dnorm(x, mean=mean(SP500.chg), sd=sd(SP500.chg)), add=TRUE, col='red')
abline(v=mean(SP500.chg), lwd=2, col='blue') # mean of S&P 500 returns
abline(v=median(SP500.chg), lwd=2, col='green') #median of S&P returns

# Graph of Q-Q plots (quantiles) for comparing S&P 500 to normal distribution
qqnorm(SP500.chg)
qqline(SP500.chg,col="red")
```


S&P 500: Return Distribution
density
0.0
1.0
2.0
-0.4
0.0
0.2
0.4
0.6
monthly % return
actual distribution
random (normal) distribution
mean actual return
median actual return

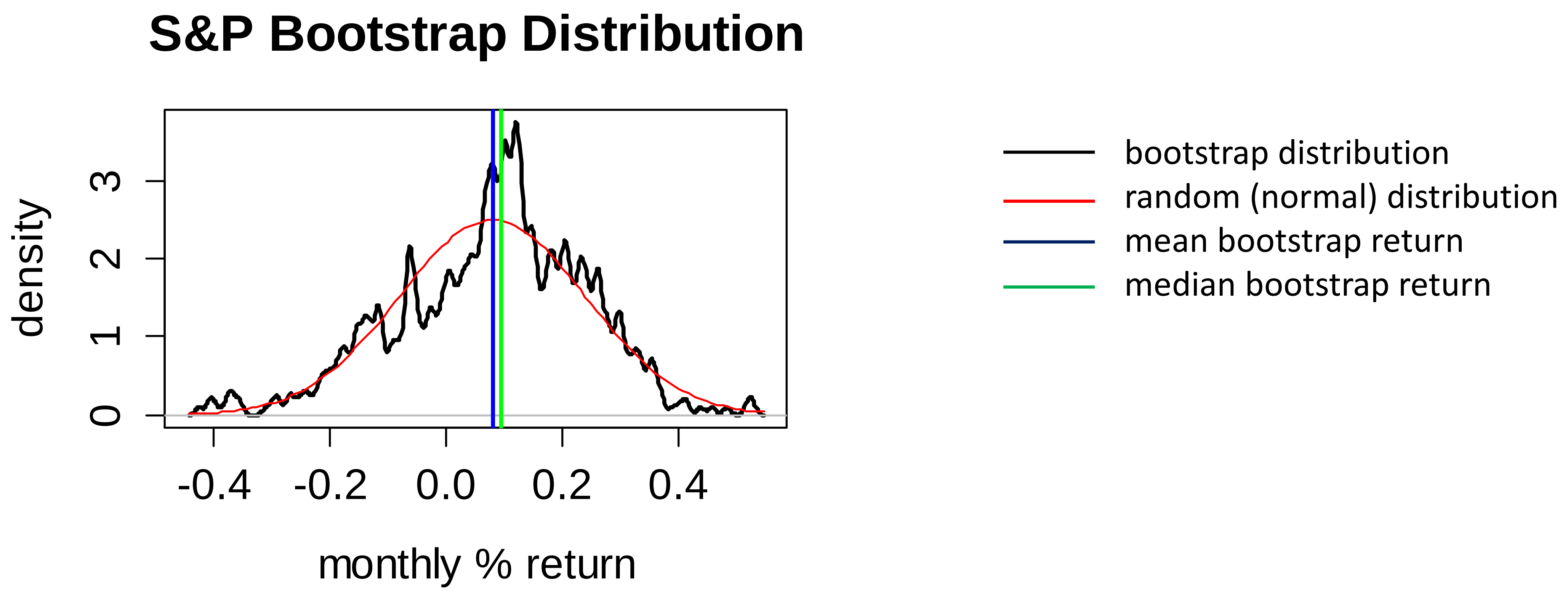


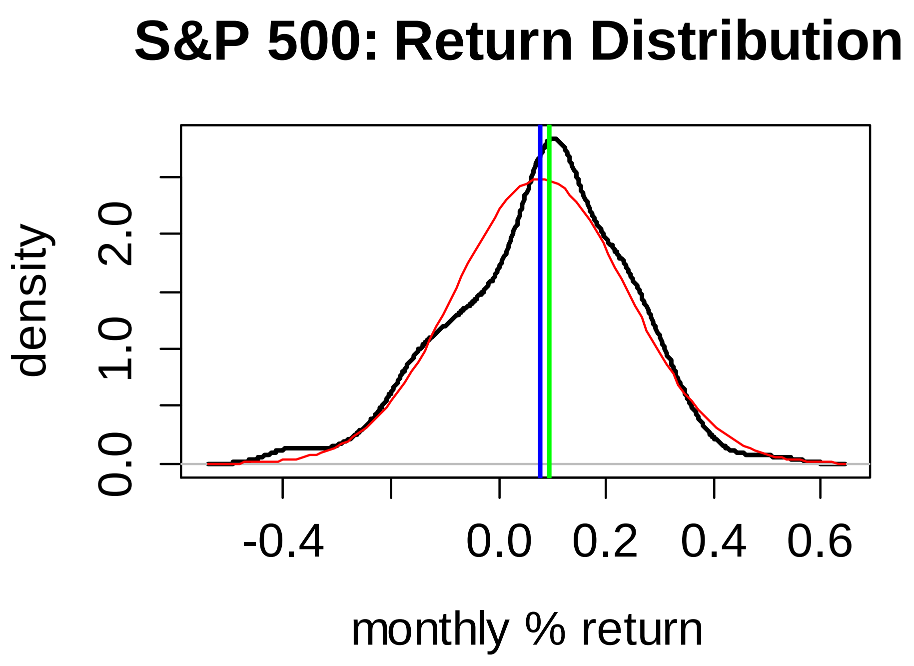
S&P 500 v. random (normal) returns

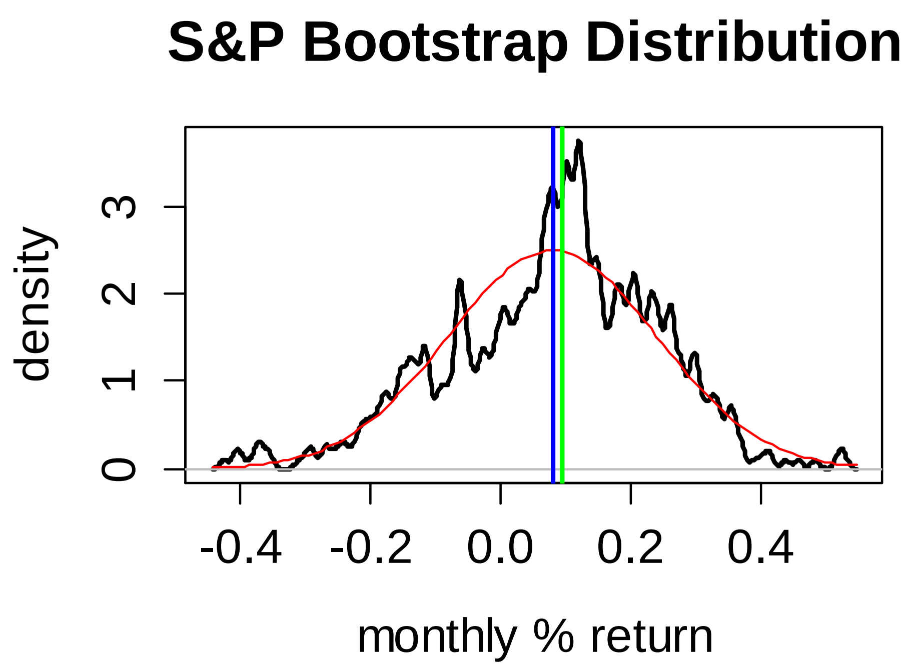
actual
random (normal)

```
# Run bootstrap on S&P 500 returns 10,000 times (by creating a list)
SP500.a <- lapply(1:10000, function(i) #run bootstrap 10000 times with list apply
sample(SP500.chg, replace = T)) # Note: replacement is TRUE

# Alternative method of bootstrapping using the "replicate" command
SP500.aa <-replicate(10000,sample(SP500.chg, replace = T))

# Adjust list for plotting by creating a single numeric data set
SP500.b <-c(do.call("cbind",SP500.a))
```

```
# Plot bootstrap data
plot(density(SP500.b), lwd=2, col="black",
    main="S&P Bootstrap Distribution",
    xlab="monthly % return",
    ylab="density")
curve(dnorm(x, mean=mean(SP500.b), sd=sd(SP500.b)), add=TRUE, col='red')
abline(v=mean(SP500.b), lwd=2, col='blue') # mean
abline(v=median(SP500.b), lwd=2, col='green') #median
```


S&P Bootstrap Distribution
density
0
1
2
3
-0.4
-0.2
0.0
0.2
0.4
monthly % return
bootstrap distribution
random (normal) distribution
mean bootstrap return
median bootstrap return


S&P 500: Return Distribution
density
0.0
1.0
2.0
-0.4
0.0
0.2
0.4
0.6
monthly % return

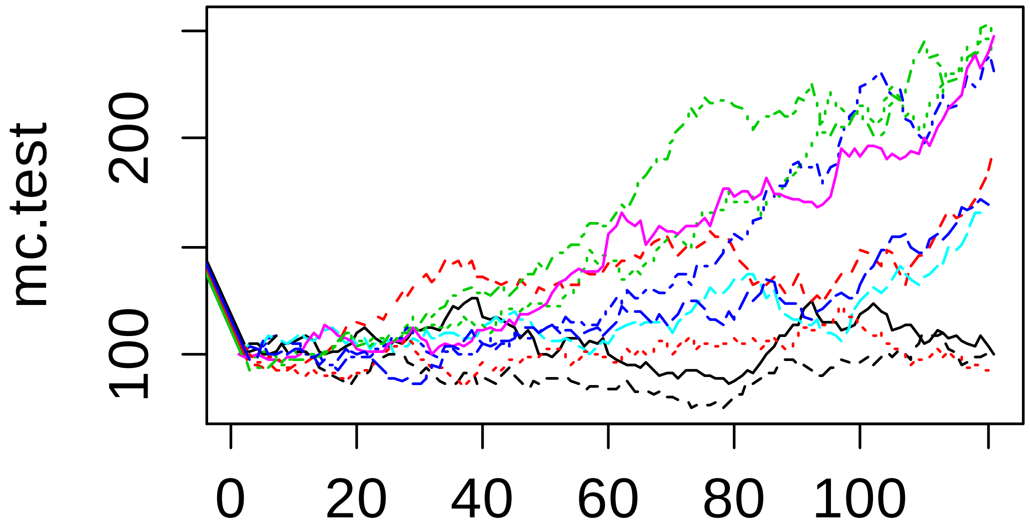

S&P Bootstrap Distribution
density
0
1
2
3
-0.4
-0.2
0.0
0.2
0.4
monthly % return

Monte Carlo Simulation

```
# Monte Carlo sim with random distribution
mc <-function(a,b,c) {
x <-c*(1/12)+(rnorm(a))*b*(sqrt(1/12))
y <- cumprod(c(100,1+x)) # 100 is the initial value
# a= number of random numbers to generate
# b=volatility assumption
# c=expected return
}
```

```
# Monte Carlo sim with student's t distribution
mc.st <-function(a,b,c,d) {
x <-c*(1/12)+(rt(a,d))*b*(sqrt(1/12))
y <- cumprod(c(100,1+x)) # 100 is the initial value
# a= number of random numbers to generate
# b=volatility assumption
# c=expected return
#  d=degrees of freedom
}
```

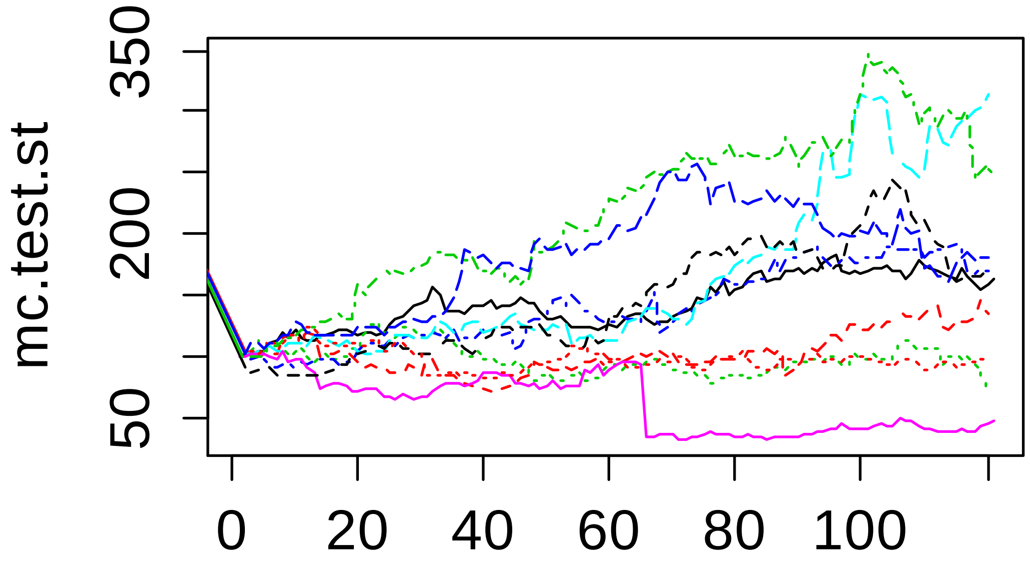
Normal Distribution Simulation



MC sim
Simulate 10 data sets
120 periods ahead
Assumptions: 10% volatilty and 5% return

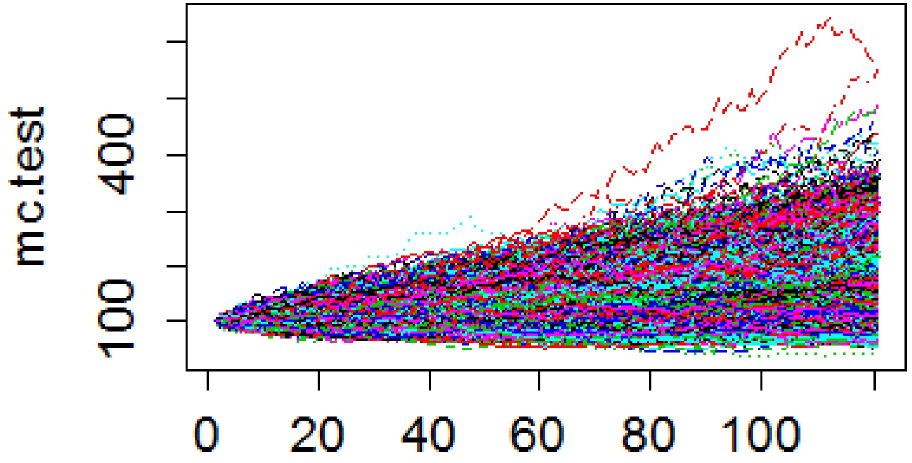
```
mc.test <-replicate(10,mc(120,.1,.05))
matplot(mc.test,type="l")
```

Student’s T Distribution Simulation
(Fat Tails)




```
# MC sim
# Simulate 10 data sets
# 120 periods ahead
# Assumptions: 10% volatilty and 5% return
 #                          and 3 degrees of freedom

mc.test.st <-replicate(10,mc.st(120,.1,.05,3))
matplot(mc.test.st,type="l")
```

10,000 simulations based on random distribution




```
mc.test <-replicate(10000,mc(120,.1,.05))
matplot(mc.test,type="l")
```

Forecasting Economic Data With ARIMA

(autoregressive integrated moving average)

via

the “forecast” package

http://cran.r-project.org/web/packages/forecast/forecast.pdf

```
#load packages
library(quantmod)
library(tseries)
library(forecast)

# Create functions: add.data.mean   NOTE: only 1-period ahead forecast code is shown
add.data.mean <-function(x) {
  x.ts <-ts(x)
  x.aa <-auto.arima(x.ts)
  x.f <-forecast(x.aa)
  x.dates <-time(x)
  x.last.date <-tail(x.dates,1)

  x.next.date.1.mean <-seq(as.Date(x.last.date), by = "month", length.out = 2)
  x.new.date.1.mean <-tail(x.next.date.1.mean,1)
  x.1.mean <- data.frame(time = seq(as.Date(x.new.date.1.mean), by = 'months', length = 1),
                        x = x.f$mean[1])
  Z.1.mean <- with(x.1.mean, xts(x, time))

  x.new.mean <-rbind(x,Z.1.mean,Z.2.mean,Z.3.mean,Z.4.mean,Z.5.mean,Z.6.mean)

}
```

```
#Create ticker file
fred.tickers <-c("USPRIV")

#Download tickers (individually)
getSymbols(fred.tickers,src="FRED")

# Create actual/forecast files
uspriv.a <-add.data.mean(USPRIV)
```

Actual data (through Aug 2013)

```
tail(USPRIV)
          USPRIV
 2013-03-01 113454
2013-04-01 113642
2013-05-01 113829
2013-06-01 114023
2013-07-01 114150
2013-08-01 114302
```

Actual data with attached ARIMA forecasts (Sep 2013 - Feb 2014)

```
tail(uspriv.a,8)
          USPRIV
2013-07-01 114150.0
2013-08-01 114302.0
2013-09-01 114443.7
2013-10-01 114585.8
2013-11-01 114730.0
2013-12-01 114856.1
2014-01-01 114993.2
2014-02-01 115127.1
```

Mean point forecasts

Thank you!

James Picerno | CapitalSpectator.com

caps@CapitalSpectator.com +1.732.710.4750