GSOC_2024_Test_MonteCarlo

Simone Mugnai

2024-02-29

Library

```
library(xts)
```

```
## Warning: package 'xts' was built under R version 4.2.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.2.2
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

Synthetic Data Generation

Generate a synthetic time series dataset to apply the bootstrap method.

```
# Set seed for reproducibility
set.seed(123)

# Generate synthetic data
dates <- seq(as.Date("2020-01-01"), by = "day", length.out = 100)
dataMatrix <- matrix(rnorm(300, mean = 100, sd = 10), ncol = 3, nrow = 100)

# Create the xts object
xtsData <- xts(dataMatrix, order.by = dates)
```

Multi-Variable Block Bootstrap Function

Define the function to perform multi-variable block bootstrap on the xts object.

```r
multiVariableBlockBootstrap <- function(data, B, blockSizeRange = c(5, 20), noiseScale = 0.1) {
  bootstrapResults <- list()
  n <- nrow(data)

  for (b in 1:B) {
    # Randomize block size within specified range
    blockSize <- sample(blockSizeRange[1]:blockSizeRange[2], 1)

    # Initialize an empty list to temporarily hold blocks
    blocks <- list()

    totalRows <- 0
    while (totalRows < n) {
      # Randomly select a block start point
      startIdx <- sample(1:(n-blockSize), 1)
      block <- data[startIdx:(startIdx+blockSize-1), ]

      # Add the block to the list
      blocks <- c(blocks, list(block))
      totalRows <- totalRows + nrow(block)

      # If the total rows exceed the original data's length, break the loop
      if (totalRows >= n) break
    }

    # Combine all blocks into one xts object
    bootstrapSample <- do.call(rbind, blocks)

    # If the bootstrap sample exceeds original length, trim it
    if (nrow(bootstrapSample) > n) {
      bootstrapSample <- bootstrapSample[1:n, ]
    }

    # Add random noise to the bootstrap sample
    noise <- matrix(rnorm(n * ncol(data), mean = 0, sd = noiseScale), nrow = n, ncol = ncol(data))
    bootstrapSample <- xts(as.matrix(bootstrapSample) + noise, order.by=index(bootstrapSample))

    # Store the bootstrap sample
    bootstrapResults[[b]] <- bootstrapSample
  }

  return(bootstrapResults)
}
```

Application and Results

Apply the bootstrap function to the synthetic xts data and examine the structure of the results.

```r
# Apply the block bootstrap function
bootstrapResults <- multiVariableBlockBootstrap(xtsData, B = 10, blockSizeRange = c(5, 20), noiseScale =

# Display the structure of the bootstrap results
str(bootstrapResults)
```

```
## List of 10
##  $ :An xts object on 2020-01-09 / 2020-03-23 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-05 / 2020-03-31 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-14 / 2020-03-19 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-02 / 2020-03-30 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-02 / 2020-03-29 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-30 / 2020-04-03 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-09 / 2020-03-30 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-02 / 2020-02-26 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-11 / 2020-04-04 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
##  $ :An xts object on 2020-01-01 / 2020-04-03 containing:
##   Data:    double [100, 3]
##   Columns: bootstrapSample.1, bootstrapSample.2, bootstrapSample.3
##   Index:   Date [100] (TZ: "UTC")
```

```
# Iterate over each bootstrap sample and display the first 5 observations for each block
lapply(bootstrapResults, function(x) head(x, 5))
```

```
## [[1]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-09          93.09939          96.24108         116.49411
## 2020-01-10          95.41328         109.24100          99.41873
## 2020-01-11         112.48677          94.36612         101.19421
## 2020-01-11         112.31865          94.13814         101.26756
```

```
## 2020-01-12         103.66713         105.96513         102.47311
##
## [[2]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-05         101.45397          90.44156          95.78243
## 2020-01-06         117.15618          99.57005          95.03958
## 2020-01-07         104.72970          92.30048          92.18608
## 2020-01-08          87.17476          83.19205          94.15783
## 2020-01-09          93.05628          96.41239         116.42013
##
## [[3]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-14         100.92883          99.49128          94.74776
## 2020-01-15          94.50057         105.17295          90.18069
## 2020-01-16         117.97879         103.03024         116.83948
## 2020-01-17         105.12307         101.07952          95.58135
## 2020-01-18          80.14131          93.46675          92.72398
##
## [[4]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-02          97.70626         102.43969         113.09094
## 2020-01-03         115.63267          97.56236          97.35171
## 2020-01-04         100.78854          96.51478         105.53530
## 2020-01-05         101.22360          90.64359          95.83279
## 2020-01-05         101.13570          90.39125          95.91698
##
## [[5]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-02          97.66747         102.68017         113.21408
## 2020-01-03         115.62851          97.51664          97.24960
## 2020-01-04         100.80399          96.48554         105.42666
## 2020-01-05         101.27449          90.62072          95.93896
## 2020-01-06         117.16703          99.66135          95.21198
##
## [[6]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-30         112.67805          99.06181          89.95299
## 2020-01-31         104.15031         114.26761         119.52171
## 2020-02-01          97.09652         104.64518          99.10140
## 2020-02-02         108.84789         100.33085         102.21066
## 2020-02-03         108.76881          95.89940          92.44937
##
## [[7]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-09          93.04369          96.11554         116.47718
## 2020-01-09          92.90164          96.09808         116.55195
## 2020-01-10          95.54573         109.21914          99.45078
## 2020-01-10          95.54866         109.16625          99.47300
## 2020-01-11         112.62599          94.22184         101.27841
##
## [[8]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-02          97.85436         102.59522         113.07934
## 2020-01-03         115.56612          97.61062          97.35390
```

```
## 2020-01-04         100.77162          96.37756         105.43164
## 2020-01-04         100.69785          96.39068         105.40222
## 2020-01-05         101.26808          90.46852          95.79324
##
## [[9]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-11         112.28465          94.18011         101.17577
## 2020-01-12         103.55872         106.05632         102.50162
## 2020-01-13         103.99547          83.78217         112.32250
## 2020-01-14         100.95154          99.25680          94.85635
## 2020-01-15          94.33928         105.15158          89.91886
##
## [[10]]
##            bootstrapSample.1 bootstrapSample.2 bootstrapSample.3
## 2020-01-01          94.49802          92.83430         122.00681
## 2020-01-02          97.63066         102.60701         113.01370
## 2020-01-03         115.42733          97.41303          97.25829
## 2020-01-04         100.82547          96.72841         105.36908
## 2020-01-05         101.46131          90.55968          95.69565
```