Calibration of Degrees of Freedom: Global Financial Cycle Project

Part-time RA for Professor Wenbin Wu

Guanxi Li

2024-07-13

1 Introduction

From the previous estimation (1. Load Data and Model Preparations, 2. Estimate the Gaussian Mode, 3. Get Posterior Draws, 4. Impulse Response Plots, 5. Produce Tables), we have already got the sample distribution of residuals for the Gaussian-errors model, say, e and e_{scaled} (residuals up to scaling by Lambda).

Next, assuming that e has an independent Student-t distribution with a degrees of freedom and unit scale, I am about to calibrate the degrees of freedom using the sample distribution of e and e_{scaled}.

2 Econometric Framework

The likelihood function of t-distribution is,

$$f(x\,|\,\theta) = \frac{\Gamma(\frac{\theta+1}{2})}{\sqrt{\theta\pi}\Gamma(\frac{\theta}{2})}(1+\frac{x^2}{\theta})^{\frac{\theta+1}{2}}$$

where θ is the degrees of freedom (DoF).

The scaled log-likelihood function of t-distribution is,

$$ll(\theta) = \sum f(e_i\,|\,\theta) = \sum \left[\log\Gamma\left(\frac{\theta+1}{2}\right) - \log\Gamma\left(\frac{\theta}{2}\right) - \frac{1}{2}\log(\theta\pi) - \frac{\theta+1}{2}\log\left(1+\frac{e^2}{\theta}\right)\right]$$

The log-likelihood function of t-distribution is,

$$ll_{scaled}(\theta) = \sum f(e_{scaled,\,i}\,|\,\theta) = \sum \left[\log\Gamma\left(\frac{\theta+1}{2}\right) - \log\Gamma\left(\frac{\theta}{2}\right) - \frac{1}{2}\log(\theta\pi) - \frac{\theta+1}{2}\log\left(1+\frac{e^2_{\text{scaled}}}{\theta}\right)\right]$$

3 Calibration of DoF using R: Boostrapping & MLE

```r
rm(list = ls())
# 1. Calibration: Bootstrapping 1.1 Setup Load the data generated from
# model_choices = 'gaussian'
load("/Users/liguanxi/Dropbox/FISF_ RA/04 Drivers of the Global Financial Cycle/GFC
 ↪  repli/Calibration_Setup.Rdata")
df <- 5.444483  # Starting point of degrees of freedom (DoF) of t-distribution

## 1.2 MLE

### 1.2.1 Log-likelihood function for unscaled residuals
```

```r
loglike <- function(theta, e) {
    if (theta <= 0)
        return(-Inf)  # Ensure theta is positive
    ll <- sum(log(gamma((theta + 1)/2)) - log(theta * pi)/2 - log(gamma(theta/2)) -
        (theta + 1)/2 * log(1 + e^2/theta))
    return(ll)
}

neg.loglike <- function(theta, e) -loglike(theta, e)

### 1.2.2 Define Wrapper Function for Optimization
neg.loglike.wrap <- function(e) {
    function(theta) {
        neg.loglike(theta, e)
    }
}

## 1.3 Bootstrapping
set.seed(2045)
n_bootstrap <- 1000  # Number of bootstrap samples
bootstrap_results <- numeric(n_bootstrap)  # Store bootstrap results

for (i in 1:n_bootstrap) {
    # Resample the data
    e_bootstrap <- sample(e, length(e), replace = TRUE)
    # Optimize using resampled data
    result <- optim(par = df, fn = neg.loglike.wrap(e_bootstrap), method = "BFGS")
    # Store the estimated parameter
    bootstrap_results[i] <- result$par
}

## 1.4 Calculate the mean and confidence interval of theta
theta_median <- mean(bootstrap_results)
theta_ci <- quantile(bootstrap_results, c(0.025, 0.975))

# 2. Calibration: MLE 2.1 Setup Load the data generated from model_choices =
# 'gaussian'
df <- theta_median  # Starting point of degrees of freedom (DoF) of t-distribution

## 2.2 MLE 2.2.1 Log-likelihood function for unscaled residuals
loglike <- function(theta) {
    if (theta <= 0)
        return(-Inf)  # Ensure theta is positive
    ll <- sum(log(gamma((theta + 1)/2)) - log(theta * pi)/2 - log(gamma(theta/2)) -
        (theta + 1)/2 * log(1 + e^2/theta))
    return(ll)
}

neg.loglike <- function(theta) -loglike(theta)

### 2.3.1 Calibration
set.seed(3045)
n_iterations <- 10  # Maximum number of MLE iterations
```

```r
mle_results <- numeric(n_iterations)  # Store MLE results
tolerance <- 1e-10  # Convergence threshold

for (i in 1:n_iterations) {
    est <- bbmle::mle2(neg.loglike, start = list(theta = df))
    new_df <- est@coef  # Use the result as the new starting point
    mle_results[i] <- new_df
    cat("Iteration", i, ": theta =", new_df, "\n")

    # Check for convergence
    if (i > 1 && abs(new_df - mle_results[i - 1]) < tolerance) {
        cat("Convergence achieved after", i, "iterations.\n")
        break
    }

    df <- new_df  # Update df for the next iteration
}
```

```
Iteration 1 : theta = 9.200688
Iteration 2 : theta = 9.200888
Iteration 3 : theta = 9.200996
Iteration 4 : theta = 9.201055
Iteration 5 : theta = 9.201086
Iteration 6 : theta = 9.201103
Iteration 7 : theta = 9.201105
Iteration 8 : theta = 9.201113
Iteration 9 : theta = 9.201113
Iteration 10 : theta = 9.201118
```

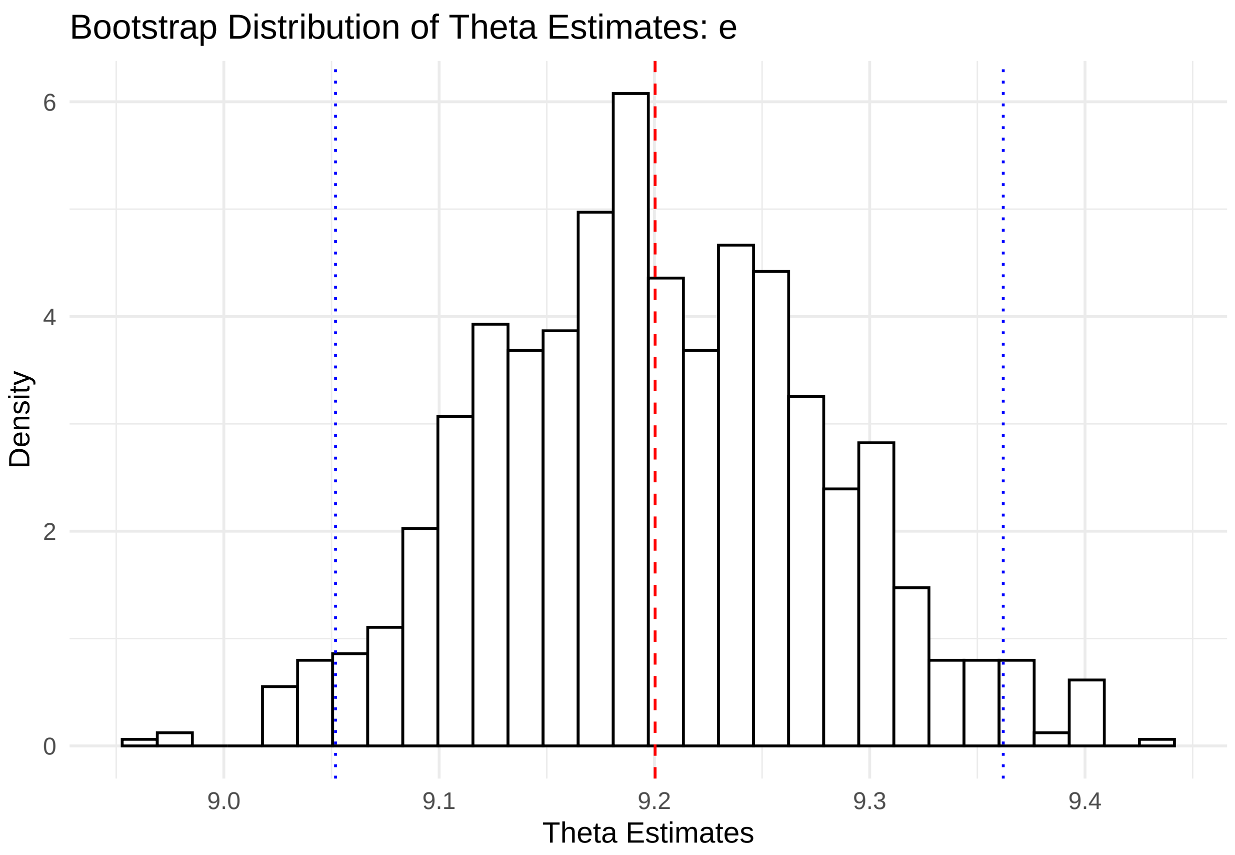
```r
## 3.1 Output: Bootstrapping 3.1.1 Median and Confidence Intervals
cat("Median of theta:", theta_median, "\n")
```

```
Median of theta: 9.200318
```

```r
cat("95% Confidence Interval of theta:", theta_ci, "\n")
```

```
95% Confidence Interval of theta: 9.051872 9.362034
```

```r
### 3.1.2 Histogram
library(ggplot2)
bootstrap_df <- data.frame(theta = bootstrap_results)
ggplot(bootstrap_df, aes(x = theta)) + geom_histogram(aes(y = after_stat(density)),
    bins = 30, color = "black", fill = "white") + labs(title = "Bootstrap Distribution of
     ↪  Theta Estimates: e",
    x = "Theta Estimates", y = "Density") + geom_vline(xintercept = theta_median,
    color = "red", linetype = "dashed") + geom_vline(xintercept = theta_ci[1], color =
     ↪  "blue",
    linetype = "dotted") + geom_vline(xintercept = theta_ci[2], color = "blue", linetype
     ↪  = "dotted") +
    theme_minimal()
```



```r
## 3.2 Output: MLE
cat("MLE Method:\n")
```

```
MLE Method:
```

```r
cat("Theta estimates across iterations:", mle_results, "\n")
```

```
Theta estimates across iterations: 9.200688 9.200888 9.200996 9.201055 9.201086 9.201103 9.201105 9.201
```

```r
cat("Final Theta estimate:", df, "\n")
```

```
Final Theta estimate: 9.201118
```