Chapter 8: Heteroskedasticity

Susumu Shikano

Last compiled at 18. Juli 2022

Simple regression model

```
n.iv <- 1
true.slope <- 0.4
true.intercept <- 0.4

x.mu <- 0.6
x.Sigma <- as.matrix(0.5)

true.err.var <- 0.05
```

We generate 1000 datasets with n=500 under the GM-assumptions. The number of independent variables is 1. The true regression line has the intercept of 0.4 and the slope of 0.4. The independent variables are generated with the mean 0.6, variances 0.5.

```
samples.srm.gm <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             het= FALSE,
                             binary.y = FALSE)

het.delta <- c(0.5,1.3)
samples.srm.het <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             het= TRUE,
                             het.delta = het.delta,
                             binary.y = FALSE)
```

Another set of samples is generated with the same parameters above, but without the homoskedasticity assumption. Here, the error variance is assumed to be:

$$var(u|x_1) = \sigma^2 \exp(\delta_0 + \delta_1 x_1)$$

with

$$\delta_0 = 0.5$$

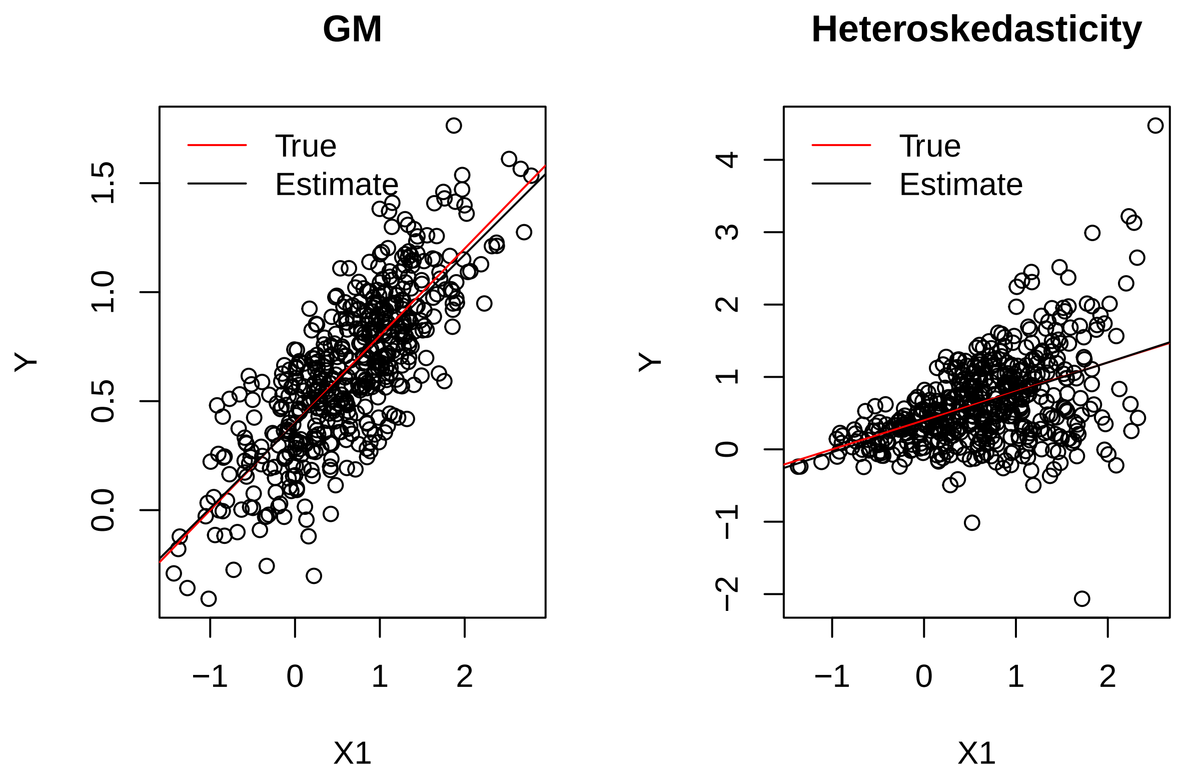
and

$$\delta_1 = 1.3$$

.

Below, we can see a sample data generated under different assumptions. The joint distribution in the right-hand side panel demonstrates an obvious heteroskedasticity.

```r
par(mfrow=c(1,2))

for (i.fig in 1:2){

  if (i.fig==1) data.1 <- samples.srm.gm$generated.data[[1]]
  if (i.fig==2) data.1 <- samples.srm.het$generated.data[[1]]

  plot(data.1$X1,data.1$y,
       xlab="X1",ylab="Y",
       main=c("GM","Heteroskedasticity")[i.fig])
  abline(reg=samples.srm.het$para,col="red")
  abline(lm(y ~ X1 ,data=data.1))
  legend("topleft",lty=1,col=c("red","black"),c("True","Estimate"),bty="n")

}
```

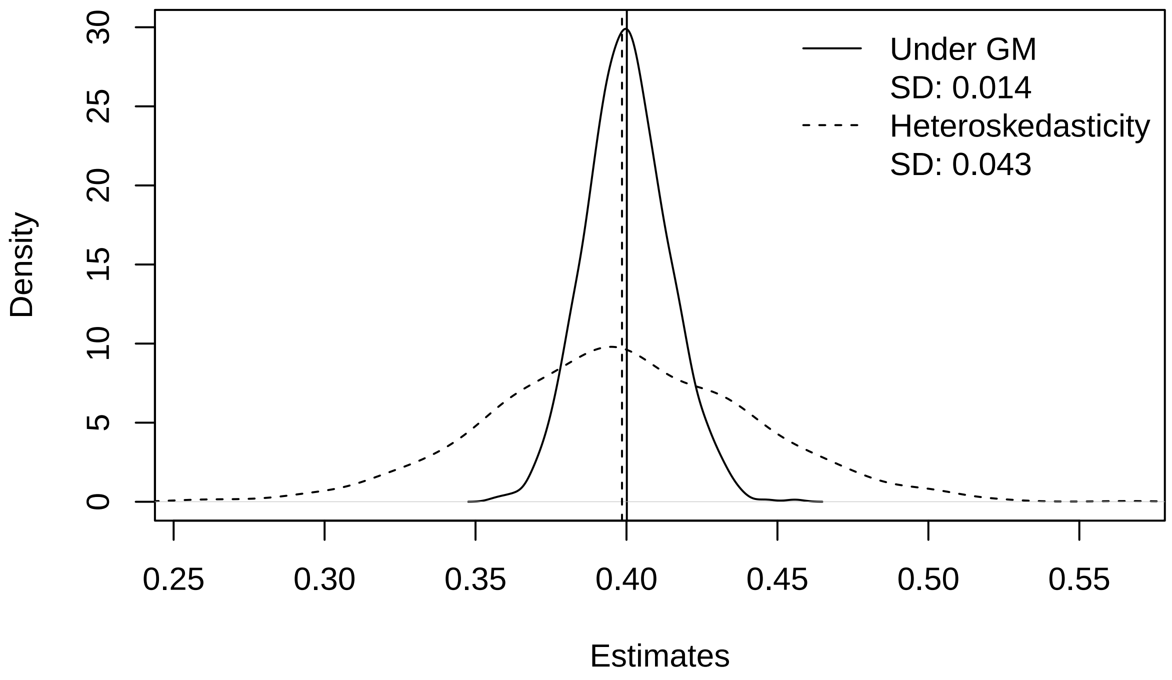


We can now estimate the regression model for each sample.

```r
all.coef <- array(NA,dim=c(num.datasets,3,2))
all.coef.se <- array(NA,dim=c(num.datasets,3,2))

for (i in 1:num.datasets){
    this.data <- samples.srm.gm$generated.data[[i]]

    lm.out <- lm(y ~ X1  ,data=  this.data)

    all.coef[i,1:2,1] <- coef(lm.out)
    all.coef[i,3,1] <- summary(lm.out)$sigma
    all.coef.se[i,c(1:2),1] <- coef(summary(lm.out))[,2]

    all.coef.se[i,3,1] <- robust.se.srm(lm.out)

    this.data <- samples.srm.het$generated.data[[i]]

    lm.out <- lm(y ~ X1  ,data=  this.data)

    all.coef[i,1:2,2] <- coef(lm.out)
    all.coef[i,3,2] <- summary(lm.out)$sigma
    all.coef.se[i,c(1:2),2] <- coef(summary(lm.out))[,2]

    all.coef.se[i,3,2] <- robust.se.srm(lm.out)
    }
dimnames(all.coef)[[2]] <- c("b0","b1","sigma")
dimnames(all.coef.se)[[2]] <- c("b0","b1","robust")
```

Below, you can see the distribution of the point estimates:

```r
    x.range <- range(c(all.coef[,"b1",]))
    plot(density.out <- density(all.coef[,"b1",1]),
         main="",xlab=paste0("Estimates"),
         xlim=x.range)
    abline(v=mean(all.coef[,"b1",1]))
    par(new=T)
    plot(density(all.coef[,"b1",2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))

    sds <- c(sqrt(naive.var(all.coef[,"b1",1])),
             sqrt(naive.var(all.coef[,"b1",2])))
    abline(v=mean(all.coef[,"b1",2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("SD:",round(sds[1],3)),
           "Heteroskedasticity",
           paste("SD:",round(sds[2],3))),bty="n")
```



If one looks at the expected values, both distributions have their expected value very close to the true value (0.4). That is, the OLS estimator is unbiased independently whether the data was generated with or without homoskedasticity.

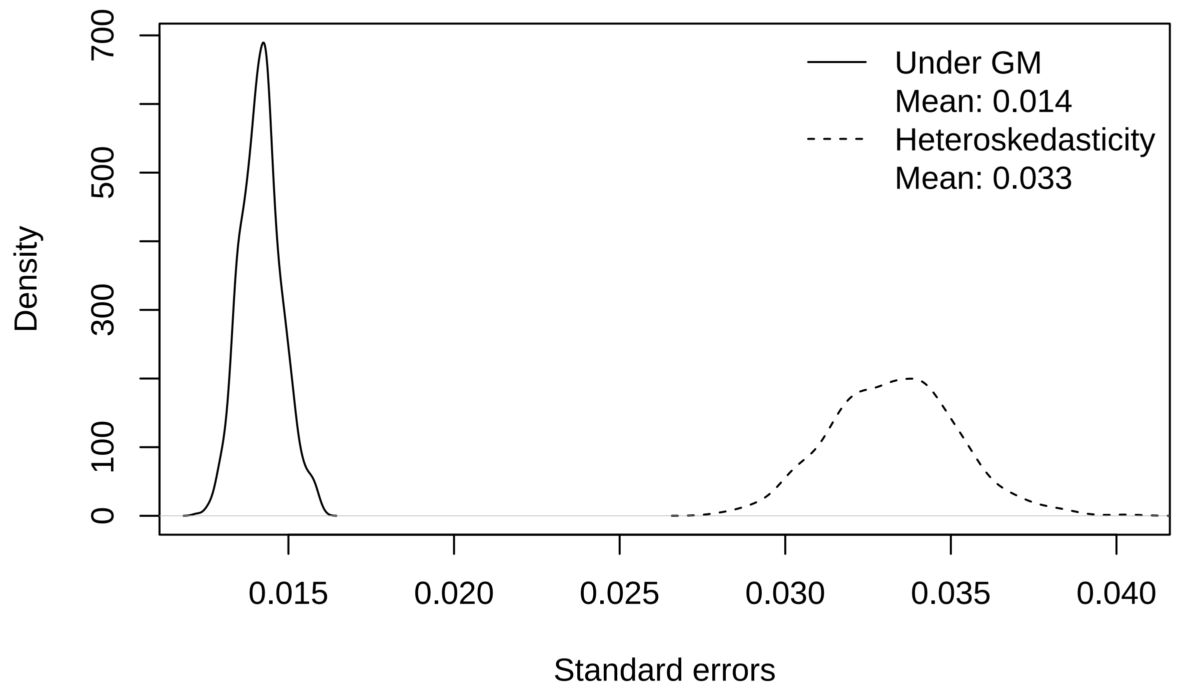
Both distribution differ in their dispersion. This is reasonable since the data under the heteroskedasticity was generated with a larger error variance

$$u^2 = \sigma^2 exp(\delta_0 + \delta_1 X1 + ...\delta_J X_j)$$

.

Important here is whether the standard errors obtained based on samples corresponds to the dispersion above. Below, you can see the distribution of standard errors:

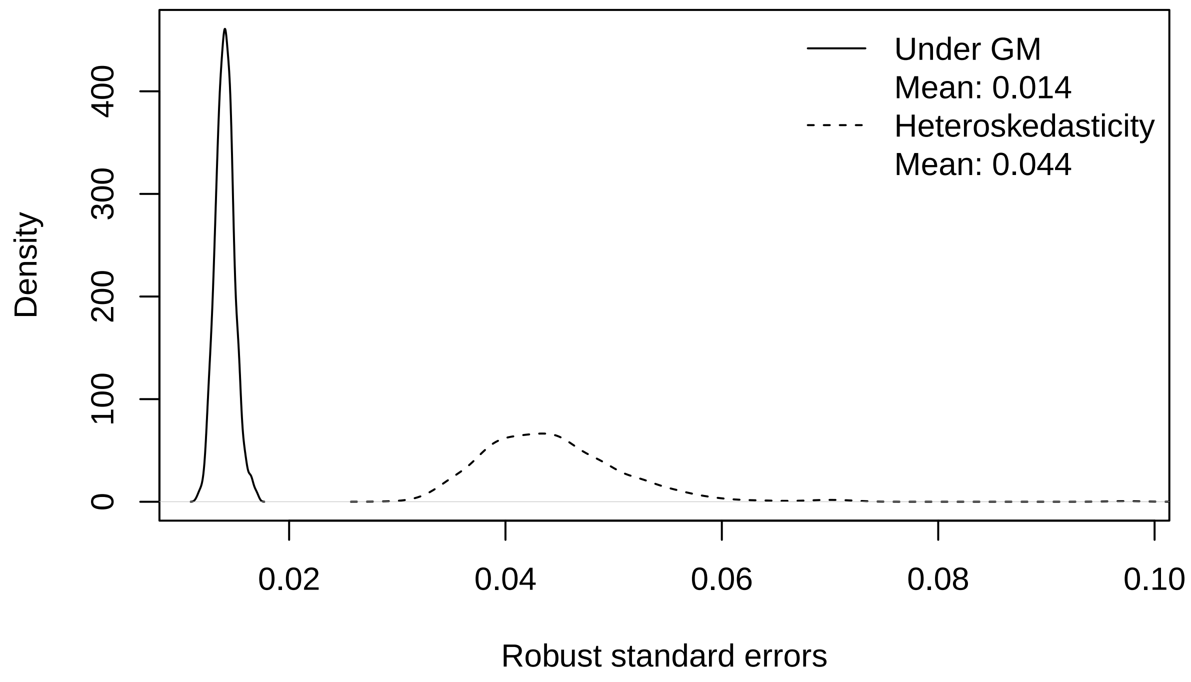
```
x.range <- range(c(all.coef.se[,"b1",]))
plot(density.out <- density(all.coef.se[,"b1",1]),
     main="",xlab=paste0("Standard errors"),
     xlim=x.range)
#abline(v=mean(all.coef.se[,"b1",1]))
par(new=T)
plot(density(all.coef.se[,"b1",2]),ann=F,xlab="",ylab="",main="",
     axes=F,
     xlim=x.range,lty=2,
     ylim=c(0,max(density.out$y)))
means <- c(mean(all.coef.se[,"b1",1]),
           mean(all.coef.se[,"b1",2]))
#abline(v=mean(all.coef[,"b1",2]),lty=2)
legend("topright",lty=c(1,NA,2,NA),
       c("Under GM",
       paste("Mean:",round(means[1],3)),
       "Heteroskedasticity",
       paste("Mean:",round(means[2],3))),bty="n")
```



Obviously, the mean standard error underestimate the dispersion of the point estimates above.

How about the robsut standard errors? Below, you will find the distributions.

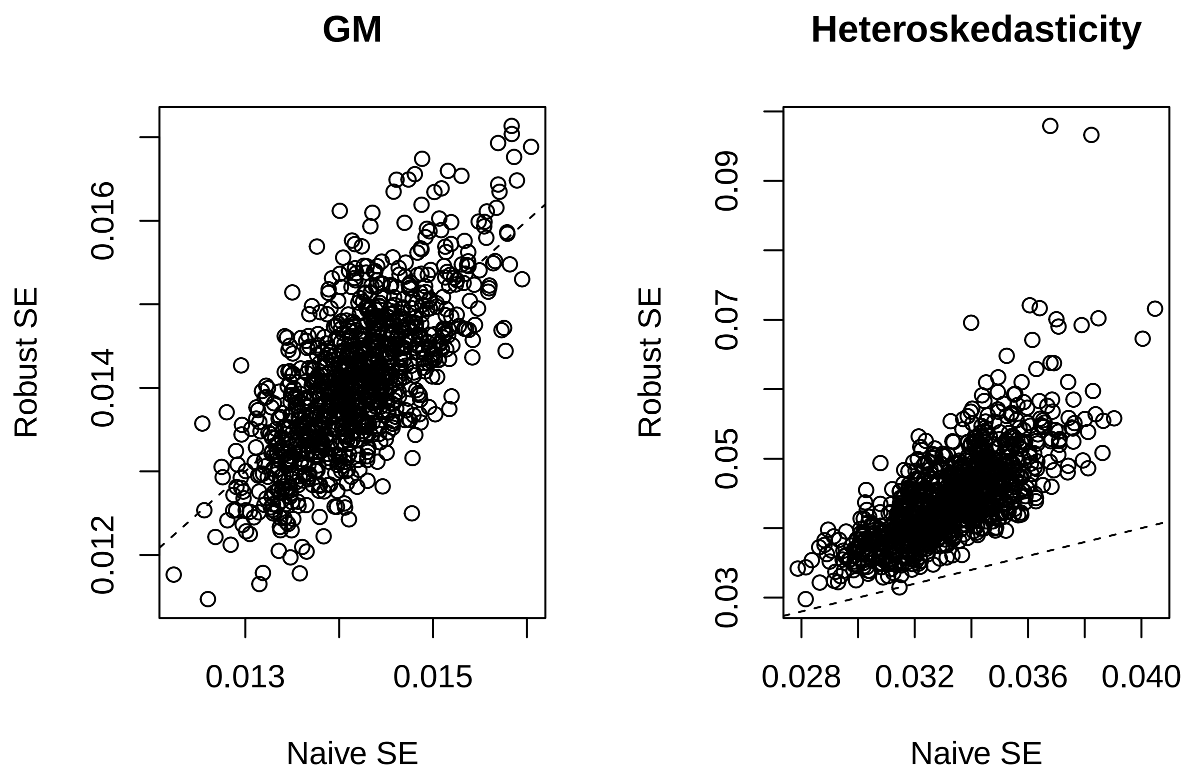
```
x.range <- range(c(all.coef.se[,"robust",]))
plot(density.out <- density(all.coef.se[,"robust",1]),
     main="",xlab=paste0("Robust standard errors"),
     xlim=x.range)
#abline(v=mean(all.coef.se[,"b1",1]))
par(new=T)
plot(density(all.coef.se[,"robust",2]),ann=F,xlab="",ylab="",main="",
     axes=F,
     xlim=x.range,lty=2,
     ylim=c(0,max(density.out$y)))
means <- c(mean(all.coef.se[,"robust",1]),
           mean(all.coef.se[,"robust",2]))
#abline(v=mean(all.coef[,"b1",2]),lty=2)
legend("topright",lty=c(1,NA,2,NA),
       c("Under GM",
       paste("Mean:",round(means[1],3)),
       "Heteroskedasticity",
       paste("Mean:",round(means[2],3))),bty="n")
```



Here, the mean of both standard errors is close to the dispersion of the point estimates above.

We can compare both types of standard errors:

```
par(mfrow=c(1,2))

for (i.fig in 1:2){

plot(all.coef.se[,"b1",i.fig],all.coef.se[,"robust",i.fig],
     ylab="Robust SE",xlab="Naive SE",
     main=c("GM","Heteroskedasticity")[i.fig])
abline(coef=c(0,1),lty=2)

}
```



Multiple regression model

```
n.iv <- 2
true.slope <- c(0.4, -0.3)
true.intercept <- 0.4

x.mu <- c(0.6,5)
x.Sigma <- cbind(c(3,-1),
                 c(-1,2))
true.err.var <- 0.05
```

We generate 1000 datasets with the same parameters above, except:

- The number of independent variables is 2.
- True slope values are 0.4, -0.3.
- The independent variables are generated with the mean 0.6, 5, and variances 3, 2 and covariance -1.

```
samples.srm.gm <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             het= FALSE,
                             binary.y = FALSE)
```

```r
het.delta <- c(0.5,1.3,0.5)
samples.srm.het <- data.generation(sample.size=sample.size,
                              n.sim=num.datasets,
                              n.iv=n.iv,
                              x.mu=x.mu,
                              x.Sigma=x.Sigma,
                              para=c(true.intercept,true.slope),
                              err.dist = "normal",
                              err.disp = true.err.var,
                              het= TRUE,
                              het.delta = het.delta,
                              binary.y = FALSE)
```

Another set of samples is generated with the same parameters above, but without the homoskedasticity assumption. Here, the error variance is assumed to be:

$$var(u|x_1) = \sigma^2 \exp(\delta_0 + \delta_1 x_1 + \delta_2 x_2)$$

with

$$\delta_0 = 0.5$$

and

$$\delta_1 = 1.3$$

and

$$\delta_2 = 0.5$$

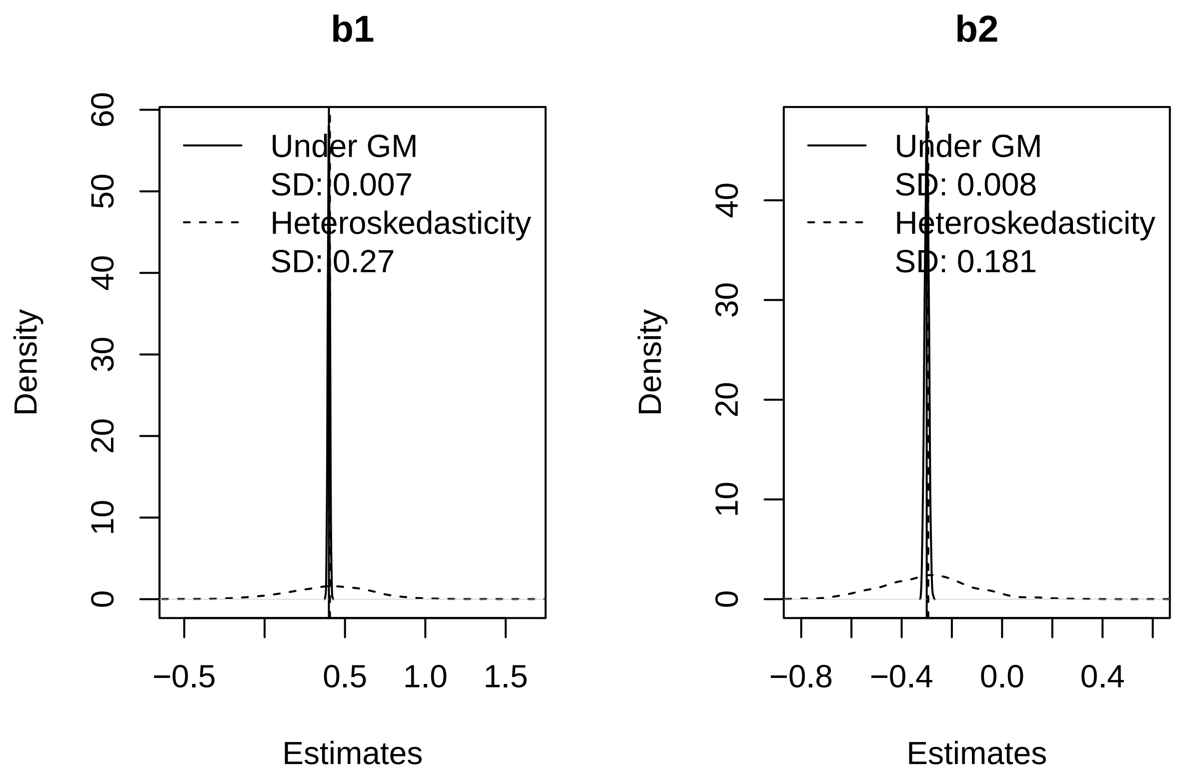
.

We can now estimate the regression model for each sample.

```r
all.coef <- array(NA,dim=c(num.datasets,4,2))
all.coef.se <- array(NA,dim=c(num.datasets,5,2))

for (i in 1:num.datasets){
    this.data <- samples.srm.gm$generated.data[[i]]

    lm.out <- lm(y ~ X1 +X2 ,data=  this.data)

    all.coef[i,1:3,1] <- coef(lm.out)
    all.coef[i,4,1] <- summary(lm.out)$sigma
    all.coef.se[i,c(1:3),1] <- coef(summary(lm.out))[,2]

    #all.coef.se[i,3,1] <- robust.se.srm(lm.out)

    this.data <- samples.srm.het$generated.data[[i]]

    lm.out <- lm(y ~ X1 +X2 ,data=  this.data)

    all.coef[i,1:3,2] <- coef(lm.out)
    all.coef[i,4,2] <- summary(lm.out)$sigma
    all.coef.se[i,c(1:3),2] <- coef(summary(lm.out))[,2]

    #all.coef.se[i,3,2] <- robust.se.srm(lm.out)
    }
dimnames(all.coef)[[2]] <- c("b0","b1","b2","sigma")
dimnames(all.coef.se)[[2]] <- c("b0","b1","b2","b1robust","b2robust")
```

Below, you can see the distribution of the point estimates:

```r
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)

    x.range <- range(c(all.coef[,target,]))
    plot(density.out <- density(all.coef[,target,1]),
         main=target,xlab=paste0("Estimates"),
         xlim=x.range)
    abline(v=mean(all.coef[,target,1]))
    par(new=T)
    plot(density(all.coef[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))

    sds <- c(sqrt(naive.var(all.coef[,target,1])),
             sqrt(naive.var(all.coef[,target,2])))
    abline(v=mean(all.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("SD:",round(sds[1],3)),
           "Heteroskedasticity",
           paste("SD:",round(sds[2],3))),bty="n")

}
```



If one looks at the expected values, both distributions have their expected value very close to the true value (0.4, -0.3). That is, the OLS estimator is unbiased independently whether the data was generated with or without homoskedasticity.

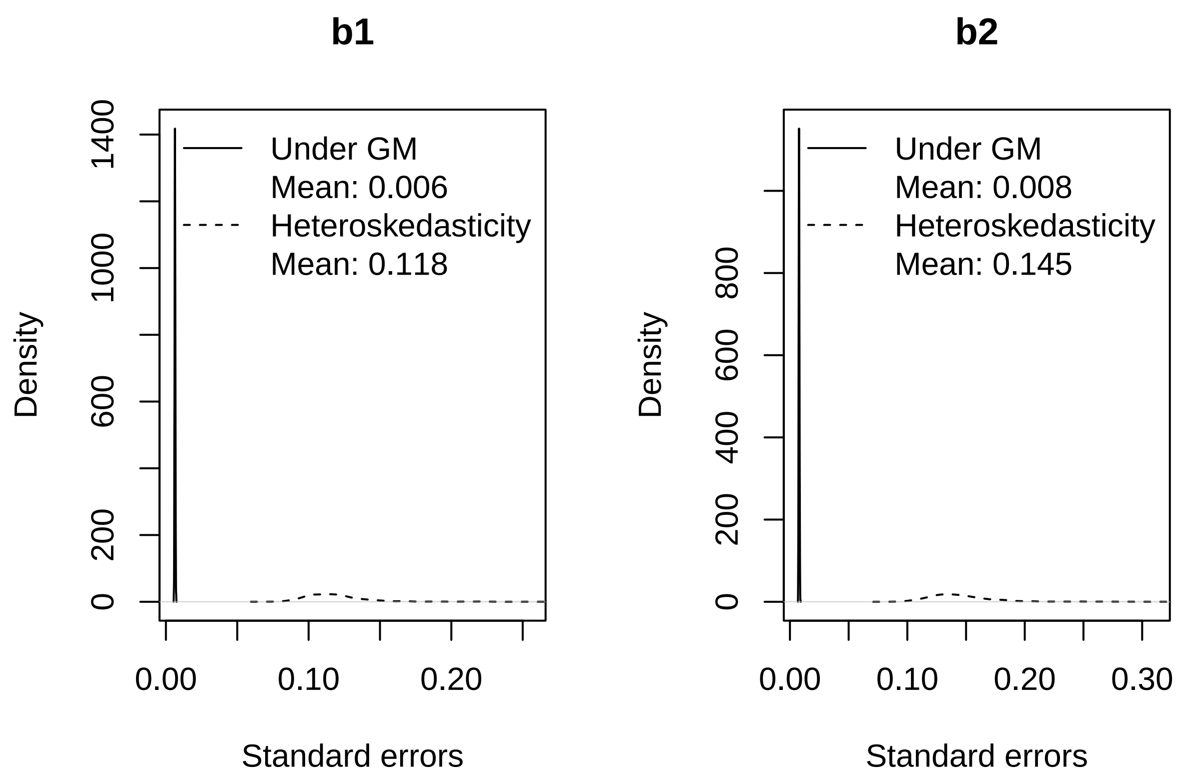
Both distribution differ in their dispersion. This is reasonable since the data under the heteroskedasticity was generated with a larger error variance

$$u^2 = \sigma^2 exp(\delta_0 + \delta_1 X1 + ...\delta_J X_j)$$

.

Important here is whether the standard errors obtained based on samples corresponds to the dispersion above. Below, you can see the distribution of standard errors:

```r
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)

    x.range <- range(c(all.coef.se[,target,]))
    plot(density.out <- density(all.coef.se[,target,1]),
         main=target,xlab=paste0("Standard errors"),
         xlim=x.range)
    #abline(v=mean(all.coef.se[,target,1]))
    par(new=T)
    plot(density(all.coef.se[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
        ylim=c(0,max(density.out$y)))
    means <- c(mean(all.coef.se[,target,1]),
               mean(all.coef.se[,target,2]))
    #abline(v=mean(all.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("Mean:",round(means[1],3)),
           "Heteroskedasticity",
           paste("Mean:",round(means[2],3))),bty="n")

}
```



Obviously, the mean standard error underestimate the dispersion of the point estimates above.

Weighted least squres (WLS)

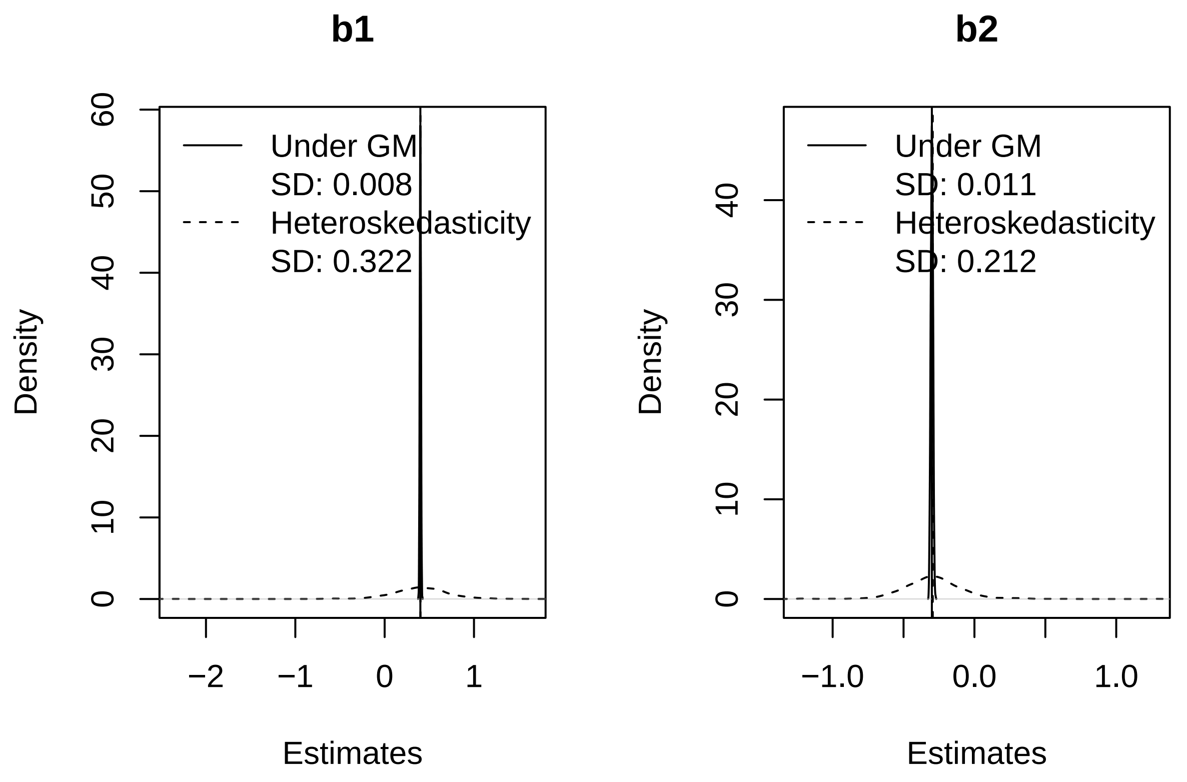
Instead of using the robust standard errors, we can compute the WLS estimates. As weight, we use the second independent variable.

```
wls.coef <- array(NA,dim=c(num.datasets,4,2))
wls.coef.se <- array(NA,dim=c(num.datasets,3,2))

for (i in 1:num.datasets){
    this.data <- samples.srm.gm$generated.data[[i]]

    this.data <- cbind(this.data,1)
    colnames(this.data)[ncol(this.data)] <- "X0"
```

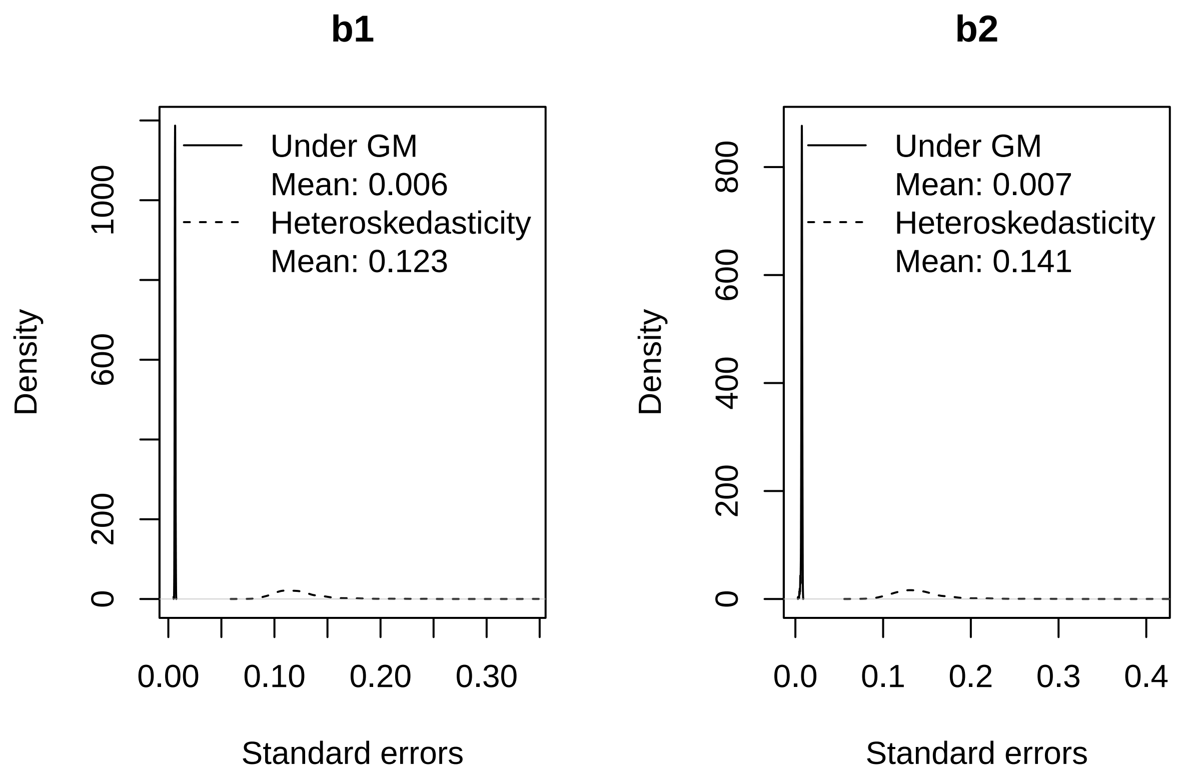
```r
    root.h <- matrix(rep(this.data$X2,ncol(this.data)),ncol=ncol(this.data))
    root.h <- sqrt(root.h)

    this.data <- this.data/root.h

    lm.out <- lm(y ~ 0 + X0 + X1 +X2 ,data=  this.data)

    wls.coef[i,1:3,1] <- coef(lm.out)
    wls.coef[i,4,1] <- summary(lm.out)$sigma
    wls.coef.se[i,c(1:3),1] <- coef(summary(lm.out))[,2]

    this.data <- samples.srm.het$generated.data[[i]]

    this.data <- cbind(this.data,1)
    colnames(this.data)[ncol(this.data)] <- "X0"

    root.h <- matrix(rep(this.data$X2,ncol(this.data)),ncol=ncol(this.data))
    root.h <- sqrt(root.h)

    this.data <- this.data/root.h

    lm.out <- lm(y ~ 0 + X0 + X1 +X2 ,data=  this.data)

    wls.coef[i,1:3,2] <- coef(lm.out)
    wls.coef[i,4,2] <- summary(lm.out)$sigma
    wls.coef.se[i,c(1:3),2] <- coef(summary(lm.out))[,2]

    }
dimnames(wls.coef)[[2]] <- c("b0","b1","b2","sigma")
dimnames(wls.coef.se)[[2]] <- c("b0","b1","b2")
```

Below, you can see the distribution of the point estimates and standard errors:

```r
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)

    x.range <- range(c(wls.coef[,target,]))
    plot(density.out <- density(all.coef[,target,1]),
         main=target,xlab=paste0("Estimates"),
         xlim=x.range)
    abline(v=mean(wls.coef[,target,1]))
    par(new=T)
    plot(density(wls.coef[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))

    sds <- c(sqrt(naive.var(wls.coef[,target,1])),
             sqrt(naive.var(wls.coef[,target,2])))
    abline(v=mean(wls.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("SD:",round(sds[1],3)),
           "Heteroskedasticity",
           paste("SD:",round(sds[2],3))),bty="n")

}
```



```
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)

    x.range <- range(c(wls.coef.se[,target,]))
    plot(density.out <- density(wls.coef.se[,target,1]),
         main=target,xlab=paste0("Standard errors"),
         xlim=x.range)
    #abline(v=mean(all.coef.se[,target,1]))
    par(new=T)
    plot(density(wls.coef.se[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))
    means <- c(mean(wls.coef.se[,target,1]),
               mean(wls.coef.se[,target,2]))
    #abline(v=mean(all.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("Mean:",round(means[1],3)),
           "Heteroskedasticity",
           paste("Mean:",round(means[2],3))),bty="n")

}
```



This performs quite poor because X2 contributes only small part of the heteroskedasticity in the data generating process.

Feasible generalized least squares (FGLS)

Instead of using a certain variable, we can now try to estimate the weight by using the OLS results.

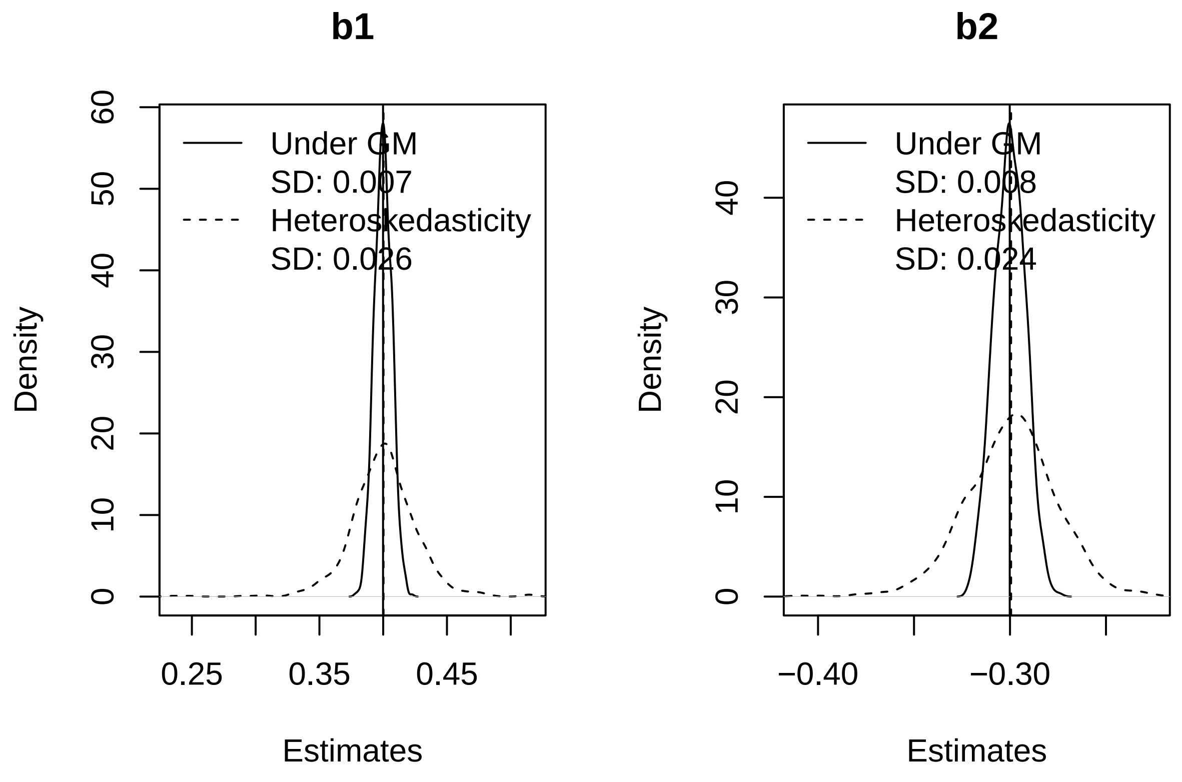
```
fgls.coef <- array(NA,dim=c(num.datasets,4,2))
fgls.coef.se <- array(NA,dim=c(num.datasets,3,2))
fgls.h.hat <- array(NA,dim=c(num.datasets,sample.size,2))

for (i in 1:num.datasets){
    this.data <- samples.srm.gm$generated.data[[i]]

    ols.out <- lm(y ~ X1 +X2 ,data=  this.data)

    this.data.2 <- this.data
    this.data.2[,1] <- log(ols.out$residuals^2)

    ols.out.2 <- lm(y ~ X1 +X2 ,data=  this.data.2)
    g.hat <- ols.out.2$fitted.values
```

```r
    h.hat <- exp(g.hat)

    this.data <- cbind(this.data,1)
    colnames(this.data)[ncol(this.data)] <- "X0"

    root.h <- matrix(rep(h.hat,ncol(this.data)),ncol=ncol(this.data))
    root.h <- sqrt(root.h)

    this.data <- this.data/root.h

    lm.out <- lm(y ~ 0 + X0 + X1 +X2 ,data=  this.data)

    fgls.coef[i,1:3,1] <- coef(lm.out)
    fgls.coef[i,4,1] <- summary(lm.out)$sigma
    fgls.coef.se[i,c(1:3),1] <- coef(summary(lm.out))[,2]
    fgls.h.hat[i,,1] <- h.hat

    this.data <- samples.srm.het$generated.data[[i]]

    ols.out <- lm(y ~ X1 +X2 ,data=  this.data)

    this.data.2 <- this.data
    this.data.2[,1] <- log(ols.out$residuals^2)

    ols.out.2 <- lm(y ~ X1 +X2 ,data=  this.data.2)
    g.hat <- ols.out.2$fitted.values
    h.hat <- exp(g.hat)

    this.data <- cbind(this.data,1)
    colnames(this.data)[ncol(this.data)] <- "X0"

    root.h <- matrix(rep(h.hat,ncol(this.data)),ncol=ncol(this.data))
    root.h <- sqrt(root.h)

    this.data <- this.data/root.h

    lm.out <- lm(y ~ 0 + X0 + X1 +X2 ,data=  this.data)

    fgls.coef[i,1:3,2] <- coef(lm.out)
    fgls.coef[i,4,2] <- summary(lm.out)$sigma
    fgls.coef.se[i,c(1:3),2] <- coef(summary(lm.out))[,2]
    fgls.h.hat[i,,2] <- h.hat

    }
dimnames(fgls.coef)[[2]] <- c("b0","b1","b2","sigma")
dimnames(fgls.coef.se)[[2]] <- c("b0","b1","b2")
```

Below, you can see the distribution of the point estimates and standard errors:

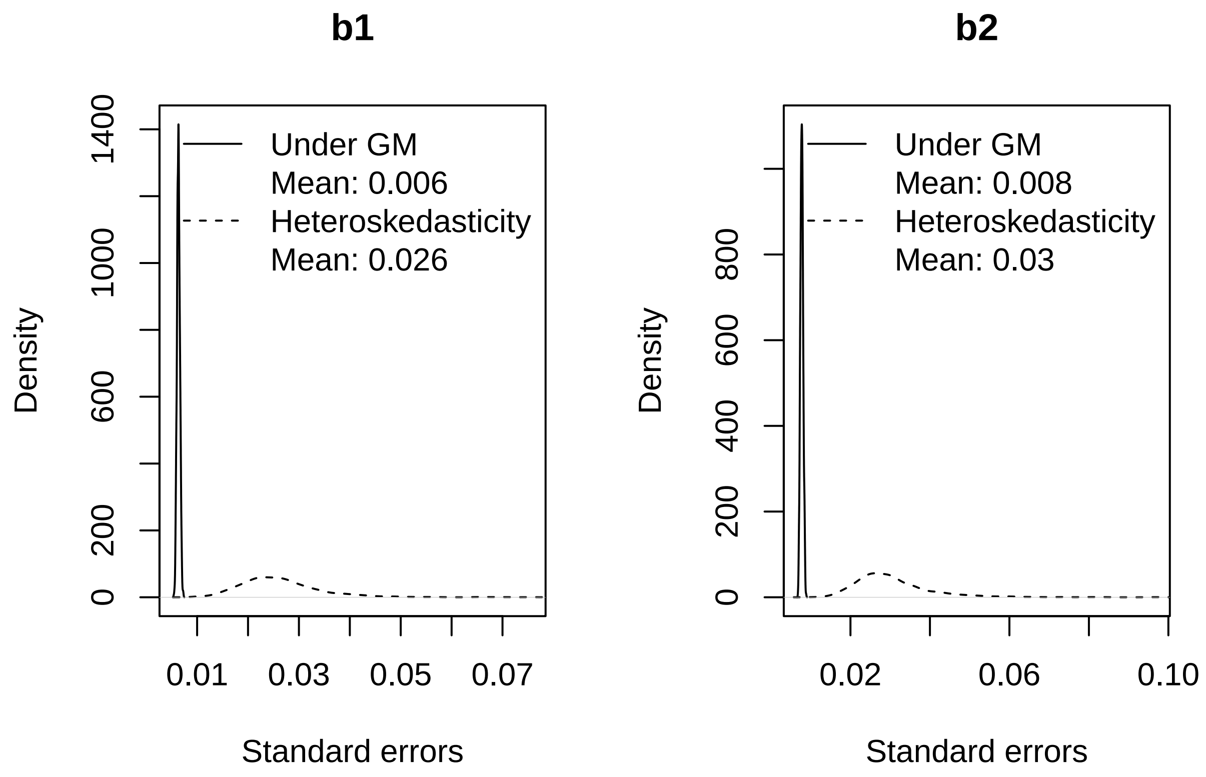
```r
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)
```

```
    x.range <- range(c(fgls.coef[,target,]))
    plot(density.out <- density(all.coef[,target,1]),
         main=target,xlab=paste0("Estimates"),
         xlim=x.range)
    abline(v=mean(fgls.coef[,target,1]))
    par(new=T)
    plot(density(fgls.coef[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))

    sds <- c(sqrt(naive.var(fgls.coef[,target,1])),
             sqrt(naive.var(fgls.coef[,target,2])))
    abline(v=mean(fgls.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("SD:",round(sds[1],3)),
           "Heteroskedasticity",
           paste("SD:",round(sds[2],3))),bty="n")

}
```



```
par(mfrow=c(1,2))
for (i.fig in 1:2){

  target <- paste0("b",i.fig)
```

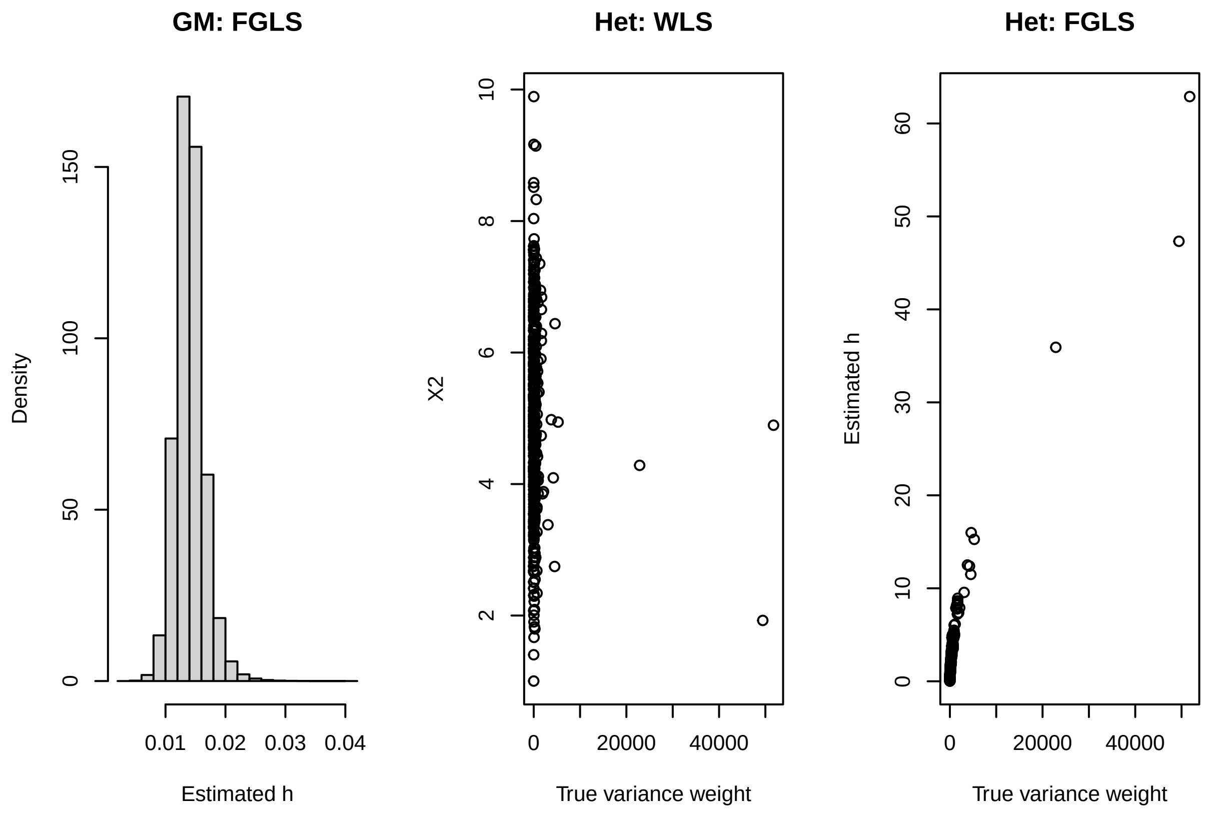
```
    x.range <- range(c(fgls.coef.se[,target,]))
    plot(density.out <- density(fgls.coef.se[,target,1]),
         main=target,xlab=paste0("Standard errors"),
         xlim=x.range)
    #abline(v=mean(all.coef.se[,target,1]))
    par(new=T)
    plot(density(fgls.coef.se[,target,2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)))
    means <- c(mean(fgls.coef.se[,target,1]),
               mean(fgls.coef.se[,target,2]))
    #abline(v=mean(all.coef[,target,2]),lty=2)
    legend("topright",lty=c(1,NA,2,NA),
           c("Under GM",
           paste("Mean:",round(means[1],3)),
           "Heteroskedasticity",
           paste("Mean:",round(means[2],3))),bty="n")

}
```



It is obvious the FGLS provides smaller variances of their estimates than WLS and OLS. That is, OLS is not the *best* estimator.

```
this.data <- samples.srm.het$generated.data[[1]]
dgp.weight <- this.data$het.weight
```

```r
par(mfrow=c(1,3))
hist(fgls.h.hat[,,1],main="GM: FGLS",xlab="Estimated h",freq=F)

wls.weight <- samples.srm.het$generated.data[[1]]$X2
plot(dgp.weight,wls.weight,main="Het: WLS",
     xlab="True variance weight",ylab="X2")


est.weight <- fgls.h.hat[,,2]
plot(dgp.weight,est.weight[1,],main="Het: FGLS",
     xlab="True variance weight",ylab="Estimated h")
```



```r
knitr::knit_exit()
```