Heteroskedasticity analysis in R

Elad Guttman

12/01/2020

As you saw in class, all the functions you need for analysis when the homoskedasticity assumption is violated are available in the `lfe` package. In this tutorial we demonstrate how to build confidence intervals and perform hypothesis testing while adjusting for heteroskedasticity standard errors, using the `lfe` package.

Building confidence intervals

Throughout this example we'll work with the 'wage2' dataset from the `wooldridge` package and use the `stargazer` package to report the results in a nice table. Let's first load all the necessary packages:

```
library(lfe)
library(wooldridge)
library(stargazer)
```

Then let's estimate the model: $lnwage = \beta_0 + \beta_1 IQ + \beta_2 black + \beta_3 married$ and report the results after adjustment for heteroskedasticity (the `felm` function uses the hinkley adjustment you learned in class):

```
data(wage2)
reg = felm(lwage ~ IQ + black + married, data = wage2)
adjusted_res = summary(reg, robust = T)
#we can now print 'adjusted_res' and see the adjusted results.
#But here we go for the preferred option - to print a nice table using stargazer.
#To do so, we need to replace the standard results with the robust results:
adjusted_se = adjusted_res$coefficients[,"Robust s.e"] #extract robust se
adjusted_pval = adjusted_res$coefficients[,"Pr(>|t|)"] #extract robust p-value
stargazer(reg, se = list(adjusted_se), p = list(adjusted_pval), type = "text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                                lwage
## -----------------------------------------------
## IQ                           0.008***
##                               (0.001)
##
## black                        -0.150***
##                               (0.039)
##
## married                      0.201***
##                               (0.041)
##
## Constant                     5.852***
##                               (0.101)
```

```
##
## ------------------------------------------------
## Observations                        935
## R2                                 0.135
## Adjusted R2                        0.132
## Residual Std. Error      0.392 (df = 931)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

Now assume we want to build a 95% confidence interval for $\theta = \beta_2 + \beta_3$. All we need for that is a point estimate (i.e. $\hat{\theta}$), and an asymptotic estimate for its variance: $V(\hat{\theta}) = V(\hat{\beta}_2) + V(\hat{\beta}_3) + 2COV(\hat{\beta}_2, \hat{\beta}_3)$. Then we can calculate $CI(\theta) = \hat{\theta} \pm 1.96\sqrt{\hat{V}(\hat{\theta})}$. You already saw how to extract coefficients from a regression object in R, so all that is left to learn is how to extract the (adjusted) covariance matrix:

```r
cov_mat = reg$robustvcv
print(cov_mat)
```

```
##               (Intercept)            IQ         black       married
## (Intercept)  1.017265e-02 -8.344884e-05 -1.543763e-03 -1.554328e-03
## IQ          -8.344884e-05  8.056285e-07  1.187003e-05  6.287421e-07
## black       -1.543763e-03  1.187003e-05  1.548976e-03  1.478548e-04
## married     -1.554328e-03  6.287421e-07  1.478548e-04  1.657885e-03
```

Now we can apply the explicit formula and report the confidence interval:

```r
theta = coef(reg)["black"] + coef(reg)["married"]
v_theta = diag(cov_mat)["black"] + diag(cov_mat)["married"] + 2*cov_mat["black", "married"]
alpha = 0.05
z = qnorm(1 - alpha/2)
cat("CI for theta: (", theta-z*sqrt(v_theta), ",", theta+z*sqrt(v_theta), ")")
```

```
## CI for theta: ( -0.06497917 , 0.1670121 )
```

Linear and non-linear hypothesis testing

Assume that in addition to reporting the the confidence interval for θ we want to test the null hypothesis: $H_0 : \theta = \beta_2 + \beta_3 = 0$. As you saw in class, we need to perform a wald-test with $r = 0$ and $R = (0, 0, 1, 1)$. Here is how we do so using the `waldtest` function, adjusting for heteroskedasticity SE[1] :

```r
r = 0
R = matrix(c(0, 0, 1, 1), nrow = 1)
waldtest(reg, R = R, r = r, type = "robust")
```

```
##           p       chi2        df1        p.F          F         df2
##   0.3886772  0.7430764  1.0000000  0.3888992  0.7430764 931.0000000
## attr(,"formula")
## ~black + married
## <environment: 0x7fe7a0c5cf58>
```

You can see from the output that the value of the test-statistic (that follows the chi-squared distribution) is 0.74, which isn't significant at the 5% level (p-value = 0.38).

[1] In class you saw the `linearhypothesis` function. Those functions are similar, but the `waldtest` function also supports testing for nonlinear hypothesis.

Next, let's see how to use `waldtest` to test the non-linear hypothesis $H_0 : g(\beta) = \beta_2^2 + \beta_3^2 = 0$. The first step is to define $g(\beta)$:

```
g_beta = function(beta){
  return(beta["black"]^2 + beta["married"]^2)
}
```

Once we have $g(\beta)$, we can provide it to the `waldtest` function, and it will test the null hypothesis (i.e., $H_0 : g(\beta) = 0$):

```
waldtest(reg, R = g_beta, type = "robust")
```

```
##            p         chi2          df1          p.F            F          df2
## 1.104345e-03 1.064390e+01 1.000000e+00 1.144513e-03 1.064390e+01 9.310000e+02
## attr(,"formula")
## function(beta){
##   return(beta["black"]^2 + beta["married"]^2)
## }
## <bytecode: 0x7fe7a2978858>
```

Here we get $W = 10.64$ and so we reject the null hypothesis (p-value $= 0.001$).