Intermediate Econometrics

Lecture 5

Vincenzo Gioia

28 October 2026

Linear Model

Limits and extension

- Although a powerful tool to explore the relationship between the covariates and the outcome, the linear model is based on the assumptions:

1. **Linearity**: The expected value of Y is a linear function of the explanatory variables $\mathbb{E}(Y_i) = \mu_i = \mathbf{x}_i^\top \beta, \quad i = 1, \ldots, n$
2. **Normality**: The variables Y_i have distribution $Y_i \sim \mathcal{N}(\mu_i, \sigma^2)$ (for the errors $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$)
3. **Homoscedasticity**: The variance of Y_i (or the error term) does not depend on i
4. **Independence**: Y_i is independent of Y_j for each pair i, j with $i \neq j$ (equivalently in terms of errors)
5. **Linear independence between explanatory variables**: the model matrix X is not stochastic and of full rank (p)

- If the assumptions do not align with the data, model-based inference can be misleading

Linear Model

- If we have evidence of **non-linearity** we can explore the following options

Transforming the predictors

$$Y_i = \beta_1 + \beta_2 g(x_{i2}) + \ldots + \beta_p g(x_{ip}) + \varepsilon_i$$

1. Several transformations (log, square root, inverse, ...)
2. Polynomial regression

Transforming the outcome

$$f(Y_i) = \beta_1 + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i^*$$

- E.g., if $f(Y_i) = \log(Y_i)$ we are assuming that $\log(Y_i)$ (and ε_i^*) are normally distributed, so Y_i and $\varepsilon_i = \exp(\varepsilon_i^*)$ are distributed as lognormal
- **Note**: We are losing the simplicity of the interpretation (the interpretation of the linear model coefficients must be done in the transformed scale)

Transforming both the predictors and the outcome

$$f(Y_i) = \beta_1 + \beta_2 g(x_{i2}) + \ldots + \beta_p g(x_{ip}) + \varepsilon_i^*$$

Linear Model

Normality

- Even without assuming the normality of errors, the OLS estimator has good properties

- However, without the normality assumption, the inferential results (confidence intervals and hypothesis tests) cannot be obtained easily

- Sometimes, we try to recover the normality assumption via transformations, for instance log()
  1. A very popular transformation is the Box-Cox transformation (we do not see it), which includes the logarithmic transformation as a special case
  2. Considering a different class of models: Generalized Linear Models (GLMs), that also include the normal linear model, where we assume that the response variable is distributed according to a certain distribution (we will see some of them)

Linear Model

Homoscedasticity

- Let's suppose that $V(\varepsilon_i) = V(Y_i) = \sigma_i^2$, for $i = 1, \ldots, n$: the variance is not constant $\implies$ **heteroscedasticity**

1. The OLS estimator $\hat{\beta}$ has mean β (unbiased)
2. The OLS estimator $\hat{\beta}$ is no longer efficient; indeed its variance-covariance matrix is

$$V(\hat{\beta}) = V((X^\top X)^{-1} X^\top Y) = (X^\top X)^{-1} X^\top V(Y) X (X^\top X)^{-1}$$

   (remember that $V(AY) = AV(Y)A^\top$)
3. The distribution is still normal $\hat{\beta} \sim \mathcal{N}(\beta, V(\hat{\beta}))$ (assuming the normality for the errors)

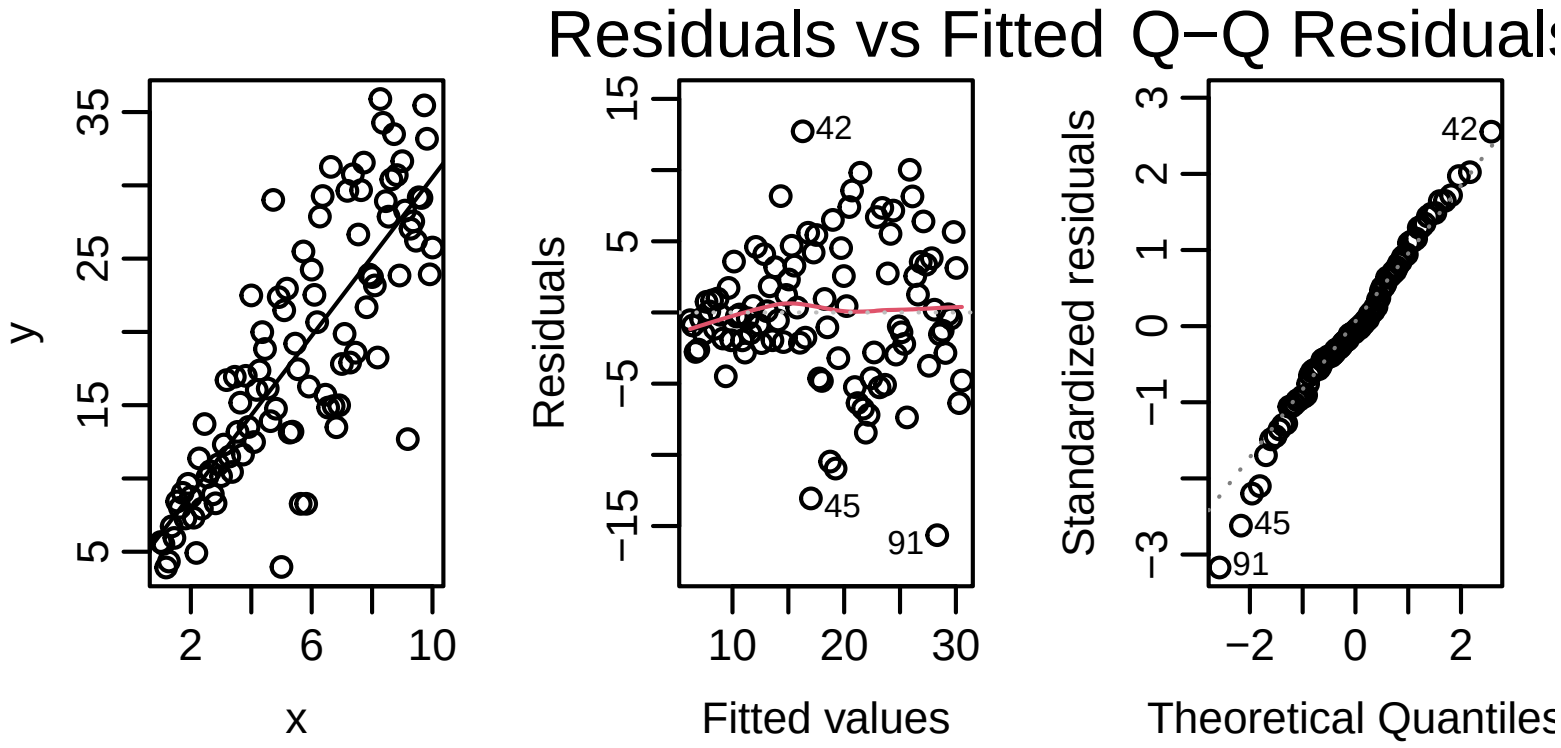
Problems

- Despite the normality, the heteroscedasticity implies that the usual procedures for obtaining hypothesis tests and confidence intervals are no longer valid

Linear Model

How to deal with heteroscedasticity

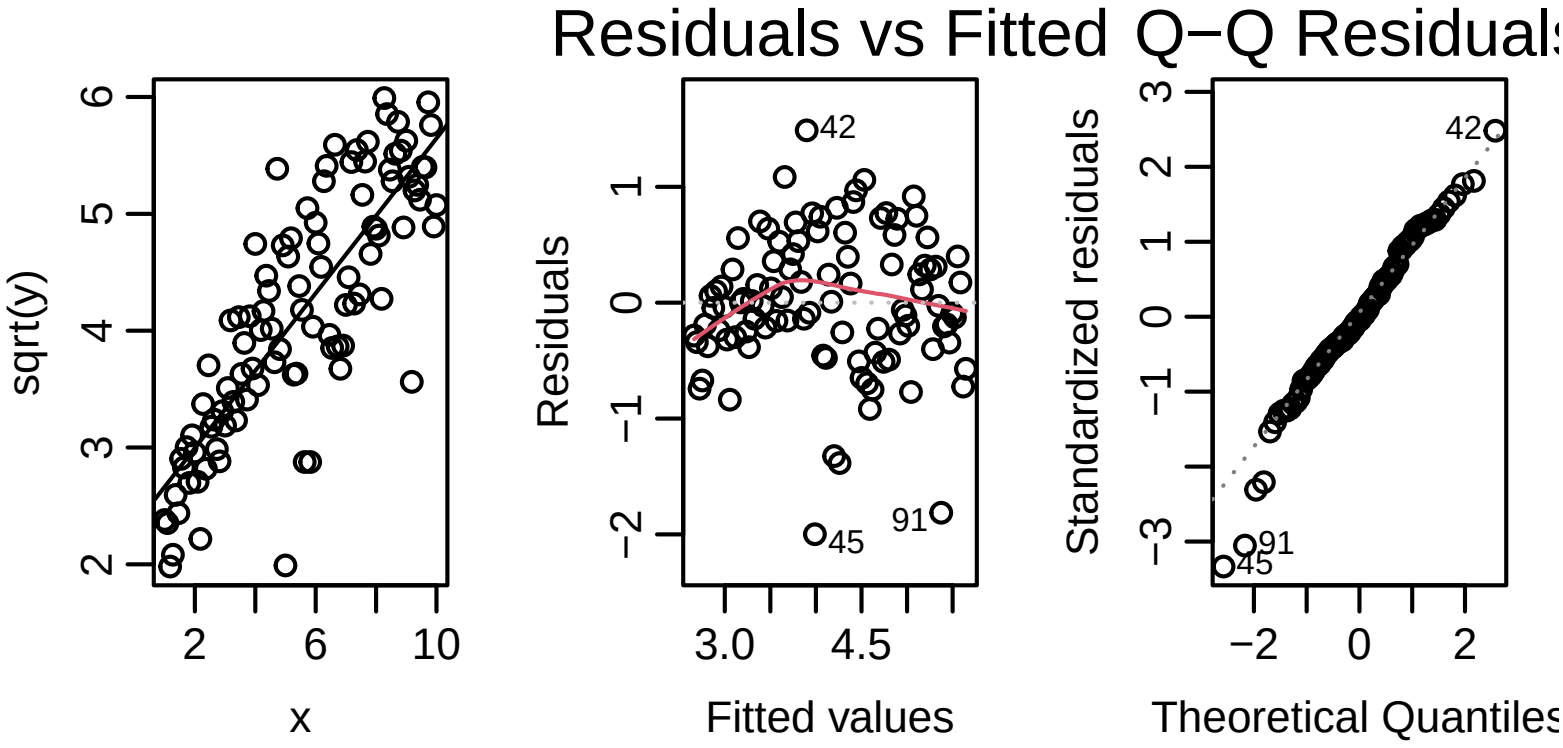
1. Transforming the outcome (there are some transformations that are able to stabilize the variance, e.g. square root, logarithm, inverse square root, arcsin of the square root). Consider the following data



Linear Model

How to deal with heteroscedasticity

1. Transforming the outcome: Consider the square root



Linear Model

Heteroscedasticity

2. Change the estimation approach: GLS (generalized least squares)

- The model structure remains the same:

$$Y = X\beta + \varepsilon$$

but we are substituting the homoscedasticity ($V(\varepsilon) = \sigma^2 I$) assumption with

$$V(\varepsilon) = \sigma^2 \Omega$$

where Ω is a known diagonal matrix (obviously not the identity)

Linear Model

Heteroscedasticity

2. Change the estimation approach: GLS (generalized least squares)

- The log-likelihood for (β, σ^2) is (up to an additive constant, $-\frac{1}{2}\log|\Omega|$, which does not depend on the parameters)

$$\ell(\beta, \sigma^2) = -\frac{n}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(y - X\beta)^\top \Omega^{-1}(y - X\beta)$$

- The maximum likelihood estimate of β is

$$\begin{aligned}\hat{\beta} &= \arg\min_{\beta}(y - X\beta)^\top \Omega^{-1}(y - X\beta) \\ &= (X^\top \Omega^{-1} X)^{-1} X^\top \Omega^{-1} y\end{aligned}$$

- So, it is easy to derive $V(\hat{\beta})$ ($\hat{\beta}$ is still normally distributed)

- Since Ω is diagonal, by using the generalized least squares we are minimizing the function

$$RSS_g(\beta) = \sum_{i=1}^{n} \frac{1}{\omega_{ii}}(y_i - x_i^\top \beta)^2$$

where ω_{ii} is the i-th diagonal element of Ω

Linear Model

Heteroscedasticity

2. Change the estimation approach: GLS (generalized least squares)

- In summary, the contributions to the sum of squares are weighted by $1/\omega_{ii}$: the greater ω_{ii} is, that is the greater the variance of the error ε_i for the i-th observation, the lower the weight associated with its contribution to the sum

- Note: If Ω is not diagonal, the generalized least squares can be used to deal with the errors' dependence, in addition to the heteroscedasticity

- The case illustrated above belongs to the class of **non-spherical disturbance errors**

Example

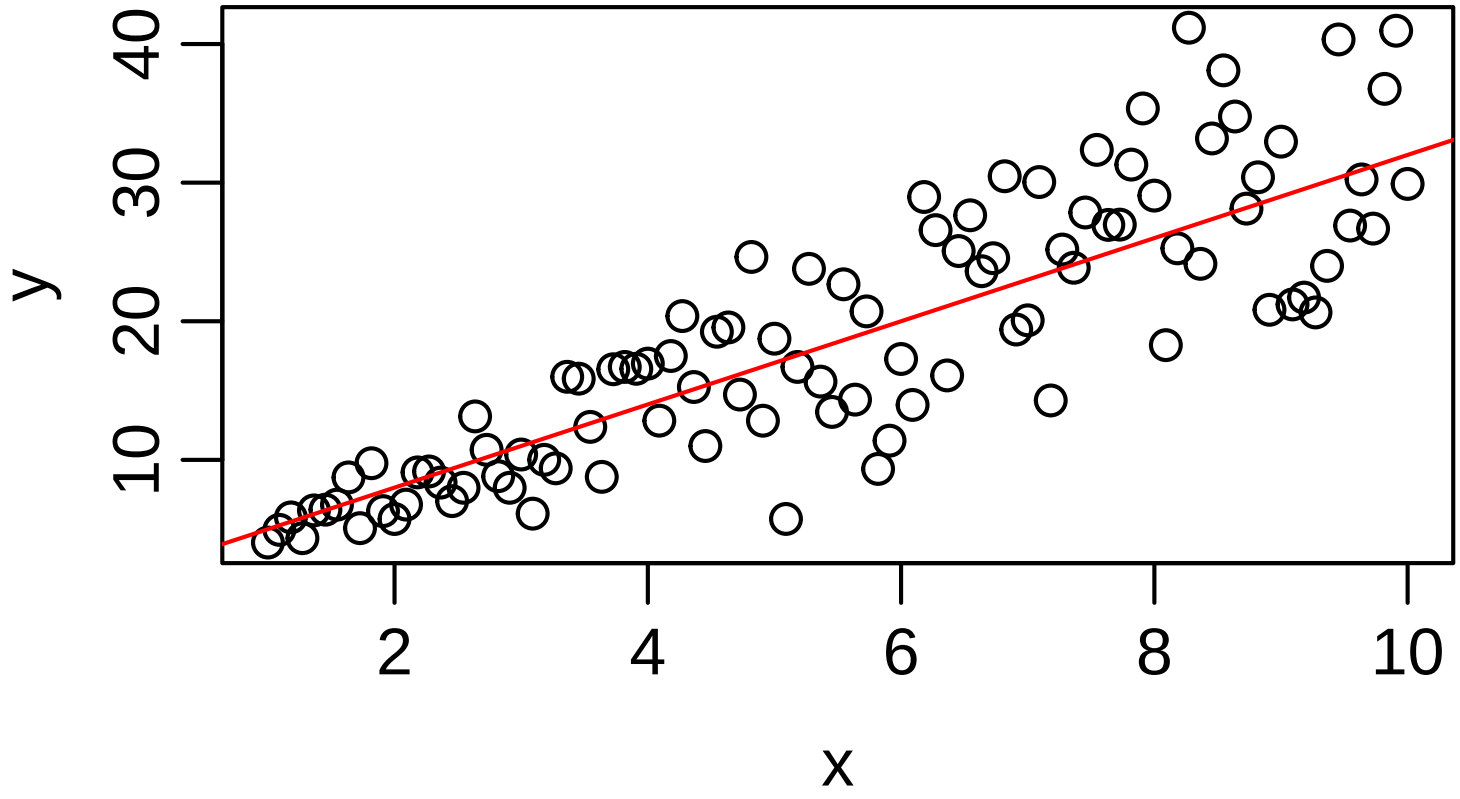
- Let's generate data assuming heteroscedasticity for the error term

```r
set.seed(3)
n <- 100
x <- seq(1, 10, length.out = n)
beta_1 <- 2; beta_2 <- 3; sigma2 <- 1
omega <- x^2  # known diagonal elements of Omega
y <- beta_1 + beta_2 * x + rnorm(n, 0, sqrt(sigma2 * omega))
```

Example

- Let's generate data assuming heteroscedasticity for the error term

```
plot(x, y)
abline(c(beta_1, beta_2), col = "red")
```



Example

- Build the negative log-likelihood function

```
nllik_gls <- function(par, y, x, omega) {
  beta_1 <- par[1]
  beta_2 <- par[2]
  sigma2 <- par[3]
  mu <- beta_1 + beta_2 * x
  n <- length(y)
  out <- n/2 * log(sigma2) + 1/(2*sigma2) * sum((y - mu)^2 / omega)
  return(out)
}
```

Example

- Let's fit the model using `nlminb()`

```
fit_gls <- nlminb(start = c(beta_1 = 0, beta_2 = 0, sigma2 = 1),
                  objective = nllik_gls, y = y, x = x,
                  omega = omega, lower = c(-Inf, -Inf, 1e-9))
fit_gls
```

```
## $par
##    beta_1    beta_2    sigma2
## 1.3700148 3.1741133 0.7107778
##
## $objective
## [1] 32.93022
##
## $convergence
## [1] 0
##
## $iterations
## [1] 13
##
## $evaluations
## function gradient
##       19       49
##
## $message
## [1] "relative convergence (4)"
```

Example

- Let's fit the model using:
  1. analytic solution;
  2. `lm` with weights;
  3. `gls()` of the R package `nlme`

```r
# 1) with the analytical solution
X <- cbind(1, x)
Om_inv <- diag(1/omega)
beta_gls <- solve(t(X) %*% Om_inv %*% X) %*% t(X) %*% Om_inv %*% y

# 2) the solution obtained using lm by specifying the weights
fit_wls <- lm(y ~ x, weights = 1/omega)

# 3) the function gls in the nlme package
library(nlme)
dat <- data.frame(y = y, x = x, omega = omega)

fit_nlme <- gls(y ~ x, data = dat,
                weights = varFixed(~ omega),    # Var(eps_i) = sigma^2 * omega_i
                method = "ML")
```

Example

- Comparisons

```
fit_ols <- lm(y~x)
cbind(nlminb       = fit_gls$par[1:2],
      closed_form = drop(beta_gls),
      lm_weights   = coef(fit_wls),
      gls_nlme    = coef(fit_nlme),
      OLS          = coef(fit_ols))
```

```
##          nlminb closed_form lm_weights gls_nlme      OLS
## beta_1 1.370015    1.370016   1.370016 1.370016 1.440915
## beta_2 3.174113    3.174113   3.174113 3.174113 3.161723
```

```
cbind(fit_gls$par[3], summary(fit_nlme)$sigma^2)
```

```
##             [,1]      [,2]
## sigma2 0.7107778 0.7107777
```

Linear Model

GLS: standard errors from the observed information

- $V(\hat{\theta}) \approx j(\hat{\theta})^{-1}$: the inverse of the Hessian of the negative log-likelihood at the maximum. Thus, we have $V(\hat{\beta}_{GLS}) = \hat{\sigma}^2 (X^\top \Omega^{-1} X)^{-1}$, with $\hat{\sigma}^2 = \frac{1}{n} \sum_i e_i^2 / \omega_{ii}$

```
se_closed <- sqrt(fit_gls$par[3] * diag(solve(t(X) %*% Om_inv %*% X)))
```

GLS: standard errors from the observed information (numerical)

- We can obtain it numerically by using the `optimHess()` function

```
H <- optimHess(fit_gls$par, nllik_gls, y = y, x = x, omega = omega)
se_hess   <- sqrt(diag(solve(H)))[1:2]
cbind(Hessian     = se_hess,
      closed_form = se_closed,
      lm_weights  = summary(fit_wls)$coefficients[, 2],
      OLS         = summary(fit_ols)$coefficients[, 2])
```

```
##          Hessian closed_form lm_weights       OLS
## beta_1 0.4371560   0.4371560  0.4415942 1.1609712
## beta_2 0.1411148   0.1411148  0.1425475 0.1905117
```

- `lm_weights` returns larger values, because it uses $n - p$ rather than n to estimate σ^2. The OLS standard errors are computed assuming homoscedasticity, so they are no longer valid.

Linear Model

OLS vs GLS: a small simulation

- Both estimators (OLS and GLS) are unbiased for β_2 under heteroscedasticity, but GLS has smaller variance

- To show this, we carry out a Monte Carlo simulation
  - generate data
  - compute the OLS and GLS estimates
  - compare them numerically and graphically (focus only on $\hat{\beta}_2$)

```
B <- 1000
est <- matrix(NA, B, 2, dimnames = list(NULL, c("OLS", "GLS")))
for(i in 1:B){
  y <- beta_1 + beta_2 * x + rnorm(n, 0, sqrt(sigma2 * omega))
  est[i, ] <-   c(OLS = coef(lm(y ~ x))[[2]],
    GLS = coef(lm(y ~ x, weights = 1/omega))[[2]])
}
```

Linear Model

OLS vs GLS: a small simulation

```
colMeans(est)
```

```
##      OLS      GLS
## 2.985467 2.994663
```

```
apply(est, 2, var)
```

```
##         OLS         GLS
## 0.06144825 0.02644579
```

```
boxplot(est, ylab = expression(hat(beta)[2]))
abline(h = beta_2, col = "red")
```

