Model Interpretation and Diagnostics

Francisco Villamil

Applied Quantitative Methods II

MA in Social Sciences, Spring 2026

Today's goals

- Move beyond coefficient tables to meaningful quantities

Today's goals

- Move beyond coefficient tables to meaningful quantities
- Compute and plot predicted values, marginal effects, and first differences

Today's goals

- Move beyond coefficient tables to meaningful quantities
- Compute and plot predicted values, marginal effects, and first differences
- Present results with publication-quality tables and coefficient plots

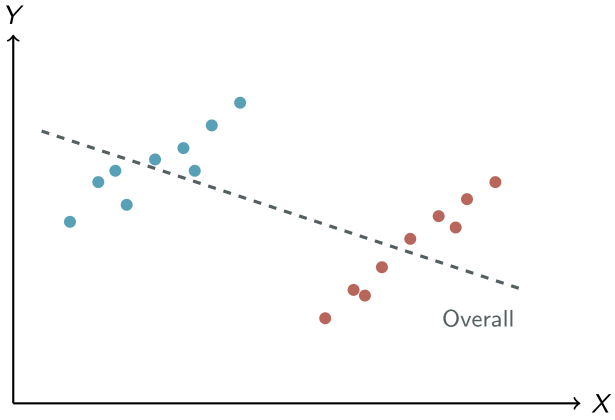
Today's goals

- Move beyond coefficient tables to meaningful quantities
- Compute and plot predicted values, marginal effects, and first differences
- Present results with publication-quality tables and coefficient plots
- Understand simulation-based uncertainty

Today's goals

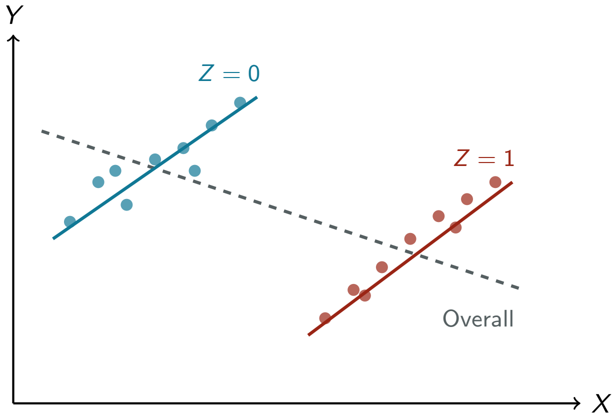
- Move beyond coefficient tables to meaningful quantities
- Compute and plot predicted values, marginal effects, and first differences
- Present results with publication-quality tables and coefficient plots
- Understand simulation-based uncertainty
- Diagnose common regression problems: heteroskedasticity, non-linearity

A motivating puzzle: Simpson's paradox



- Overall trend is **negative**, but within each group it's **positive**

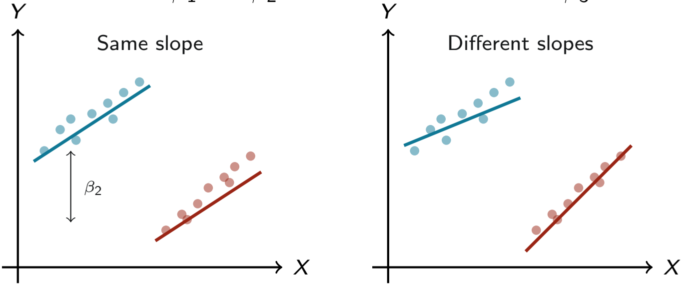
A motivating puzzle: Simpson's paradox



- Overall trend is **negative**, but within each group it's **positive**
- Ignoring Z gives the **wrong answer** — we need to account for it

Controlling vs. interacting

Control: $Y = \beta_1 X + \beta_2 Z$ **Interaction:** $+\beta_3 X \cdot Z$



- **Controlling**: adjusts the level (intercept) — the effect of X stays the same
- **Interacting**: allows the **slope** of X to differ across groups

Roadmap

Beyond Coefficient Tables

Predicted Values and Marginal Effects

Presenting Results

Diagnostics

Wrap-up

The interpretation problem

- We know how to estimate models and read output

The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?

The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?
- Raw coefficients are often not directly informative:

The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?
- Raw coefficients are often not directly informative:
  - → Logit: coefficients are in log-odds (last week)

The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?
- Raw coefficients are often not directly informative:
  - → Logit: coefficients are in log-odds (last week)
  - → Interactions: the coefficient on X_1 depends on X_2

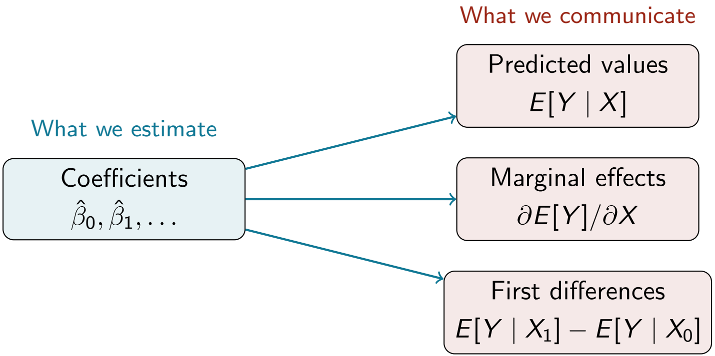
The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?
- Raw coefficients are often not directly informative:
  - → Logit: coefficients are in log-odds (last week)
  - → Interactions: the coefficient on X_1 depends on X_2
  - → Log transformations: coefficients are elasticities or semi-elasticities

The interpretation problem

- We know how to estimate models and read output
- But what do the numbers actually **mean**?
- Raw coefficients are often not directly informative:
  - → Logit: coefficients are in log-odds (last week)
  - → Interactions: the coefficient on X_1 depends on X_2
  - → Log transformations: coefficients are elasticities or semi-elasticities
  - → Different scales: is $\hat{\beta} = 0.003$ big or small?

Coefficients vs. quantities of interest



You estimate a model with GDP (in dollars), population (in millions), and an interaction between them.

What does the coefficient on GDP tell you?

Roadmap

Beyond Coefficient Tables

Predicted Values and Marginal Effects

Presenting Results

Diagnostics

Wrap-up

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: “What does the model predict for this scenario?”

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: "What does the model predict for this scenario?"
- Set predictors to specific values, compute $\hat{Y}$

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: "What does the model predict for this scenario?"

- Set predictors to specific values, compute $\hat{Y}$
- Example: "What is predicted income for a 35-year-old with a college degree?"

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: "What does the model predict for this scenario?"

- Set predictors to specific values, compute $\hat{Y}$
- Example: "What is predicted income for a 35-year-old with a college degree?"

- In R:

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: "What does the model predict for this scenario?"

- Set predictors to specific values, compute $\hat{Y}$
- Example: "What is predicted income for a 35-year-old with a college degree?"

- In R:
  → `predictions(model)`

Predicted values: $E[Y \mid X]$

- The most intuitive quantity: "What does the model predict for this scenario?"

- Set predictors to specific values, compute $\hat{Y}$
- Example: "What is predicted income for a 35-year-old with a college degree?"

- In R:
  - → `predictions(model)`
  - → `predictions(model,`

```
        newdata = datagrid(age = 35, college = 1))
```

Plotting predictions

- The gold standard for communicating model results

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age
  - → Includes 95% confidence band (for the mean, not individual predictions)

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age
  - → Includes 95% confidence band (for the mean, not individual predictions)
  - → Holds other variables at their means (or modes)

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age
  - → Includes 95% confidence band (for the mean, not individual predictions)
  - → Holds other variables at their means (or modes)
- With two conditions:

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age
  - → Includes 95% confidence band (for the mean, not individual predictions)
  - → Holds other variables at their means (or modes)
- With two conditions:
  - →
```
plot_predictions(model,
      condition = c("age", "female"))
```

Plotting predictions

- The gold standard for communicating model results
- `plot_predictions(model, condition = "age")`
  - → Shows $\hat{Y}$ across values of age
  - → Includes 95% confidence band (for the mean, not individual predictions)
  - → Holds other variables at their means (or modes)
- With two conditions:
  - →
    ```
    plot_predictions(model,
          condition = c("age", "female"))
    ```
  - → Separate lines/panels for each group

Marginal effects

$$\text{Marginal effect of } X_1 = \frac{\partial E[Y \mid X]}{\partial X_1}$$

- "How much does Y change for a small change in X_1?"

Marginal effects

$$\text{Marginal effect of } X_1 = \frac{\partial E[Y \mid X]}{\partial X_1}$$

- "How much does Y change for a small change in X_1?"
- For OLS without interactions: marginal effect $= \hat{\beta}_1$ (constant)

Marginal effects

$$\text{Marginal effect of } X_1 = \frac{\partial E[Y \mid X]}{\partial X_1}$$

- "How much does Y change for a small change in X_1?"
- For OLS without interactions: marginal effect $= \hat{\beta}_1$ (constant)
- For logit, interactions, polynomials: marginal effect **varies**

Marginal effects

$$\text{Marginal effect of } X_1 = \frac{\partial E[Y \mid X]}{\partial X_1}$$

- "How much does Y change for a small change in X_1?"
- For OLS without interactions: marginal effect $= \hat{\beta}_1$ (constant)
- For logit, interactions, polynomials: marginal effect **varies**
- **Average marginal effect (AME)**: average across all observations

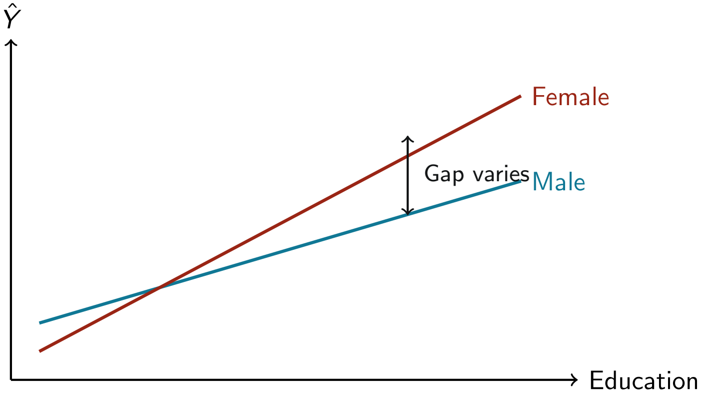
Marginal effects

$$\text{Marginal effect of } X_1 = \frac{\partial E[Y \mid X]}{\partial X_1}$$

- "How much does Y change for a small change in X_1?"
- For OLS without interactions: marginal effect $= \hat{\beta}_1$ (constant)
- For logit, interactions, polynomials: marginal effect **varies**
- **Average marginal effect (AME)**: average across all observations
  - $\rightarrow$ `avg_slopes(model)`

Marginal effects with interactions

$$Y = \beta_0 + \beta_1 \text{Education} + \beta_2 \text{Female} + \beta_3 (\text{Education} \times \text{Female}) + \varepsilon$$



- ME of Education for males: β_1
- ME of Education for females: $\beta_1 + \beta_3$

Conditional marginal effects in R

- Marginal effect of X_1 at specific values of X_2:

Conditional marginal effects in R

- Marginal effect of X_1 at specific values of X_2:
  - → `slopes(model, variables = "education",`
    `newdata = datagrid(female = c(0, 1)))`

Conditional marginal effects in R

- Marginal effect of X_1 at specific values of X_2:
  →
```
slopes(model, variables = "education",
       newdata = datagrid(female = c(0, 1)))
```
- Plot how the marginal effect varies:

Conditional marginal effects in R

- Marginal effect of X_1 at specific values of X_2:
  →
  ```
  slopes(model, variables = "education",
         newdata = datagrid(female = c(0, 1)))
  ```
- Plot how the marginal effect varies:
  →
  ```
  plot_slopes(model, variables = "education",
         condition = "female")
  ```

Conditional marginal effects in R

- Marginal effect of X_1 at specific values of X_2:

  → `slopes(model, variables = "education",`
  `newdata = datagrid(female = c(0, 1)))`

- Plot how the marginal effect varies:

  → `plot_slopes(model, variables = "education",`
  `condition = "female")`

- This replaces the old manual approach of computing $\beta_1 + \beta_3 \cdot X_2$ by hand

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - $\rightarrow$ $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - $\rightarrow$ $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - $\rightarrow$ $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:
  - $\rightarrow$
```
comparisons(model,
      variables = list(college = c(0, 1)))
```

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - → $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:
  - → `comparisons(model,`
    `variables = list(college = c(0, 1)))`
  - → `avg_comparisons(model, variables = "college")`

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - $\rightarrow$ $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:
  - $\rightarrow$
    ```
    comparisons(model,
          variables = list(college = c(0, 1)))
    ```
  - $\rightarrow$
    ```
    avg_comparisons(model, variables = "college")
    ```
- Especially useful for:

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - → $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:
  - →
    ```
    comparisons(model,
          variables = list(college = c(0, 1)))
    ```
  - → `avg_comparisons(model, variables = "college")`
- Especially useful for:
  - → Binary/categorical predictors

First differences (comparisons)

- "How different is $\hat{Y}$ between two scenarios?"
- Example: predicted income for college vs. no college
  - → $\Delta = E[Y \mid \text{college} = 1] - E[Y \mid \text{college} = 0]$
- In R:
  - →
    ```
    comparisons(model,
          variables = list(college = c(0, 1)))
    ```
  - →
    ```
    avg_comparisons(model, variables = "college")
    ```
- Especially useful for:
  - → Binary/categorical predictors
  - → Comparing meaningful scenarios (e.g., min vs. max)

Choosing the right quantity

Quantity	Question	R function
Predicted value	What does the model predict here?	`predictions()`
Marginal effect	How much does Y change per unit of X?	`slopes()`
Average ME	What is the average effect across the sample?	`avg_slopes()`
First difference	How different are two scenarios?	`comparisons()`

- All from the `marginaleffects` package (Arel-Bundock et al.)

Choosing the right quantity

Quantity	Question	R function
Predicted value	What does the model predict here?	`predictions()`
Marginal effect	How much does Y change per unit of X?	`slopes()`
Average ME	What is the average effect across the sample?	`avg_slopes()`
First difference	How different are two scenarios?	`comparisons()`

- All from the `marginaleffects` package (Arel-Bundock et al.)
- All compute standard errors and CIs automatically

Choosing the right quantity

Quantity	Question	R function
Predicted value	What does the model predict here?	`predictions()`
Marginal effect	How much does Y change per unit of X?	`slopes()`
Average ME	What is the average effect across the sample?	`avg_slopes()`
First difference	How different are two scenarios?	`comparisons()`

- All from the `marginaleffects` package (Arel-Bundock et al.)
- All compute standard errors and CIs automatically
- All work with `lm()`, `glm()`, and many other model types

Roadmap

Beyond Coefficient Tables

Predicted Values and Marginal Effects

Presenting Results

Diagnostics

Wrap-up

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs
  - → `coef_rename` to clean up variable names

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs
  - → `coef_rename` to clean up variable names
  - → `gof_map` to select goodness-of-fit statistics

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs
  - → `coef_rename` to clean up variable names
  - → `gof_map` to select goodness-of-fit statistics
- Multiple models side by side:

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs
  - → `coef_rename` to clean up variable names
  - → `gof_map` to select goodness-of-fit statistics
- Multiple models side by side:
  - → `modelsummary(list("OLS" = m1, "Logit" = m2))`

Publication-quality tables with `modelsummary()`

- We already know `modelsummary()` from Session 2
- Key options:
  - → `stars = TRUE` for significance stars
  - → `vcov = "robust"` for robust SEs
  - → `coef_rename` to clean up variable names
  - → `gof_map` to select goodness-of-fit statistics
- Multiple models side by side:
  - → `modelsummary(list("OLS" = m1, "Logit" = m2))`
- Output to LaTeX, Word, HTML, or the console

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = “no effect”

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = "no effect"
- When are coefficient plots better than tables?

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = "no effect"
- When are coefficient plots better than tables?
  - → Many predictors

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = "no effect"
- When are coefficient plots better than tables?
  - → Many predictors
  - → Comparing across model specifications

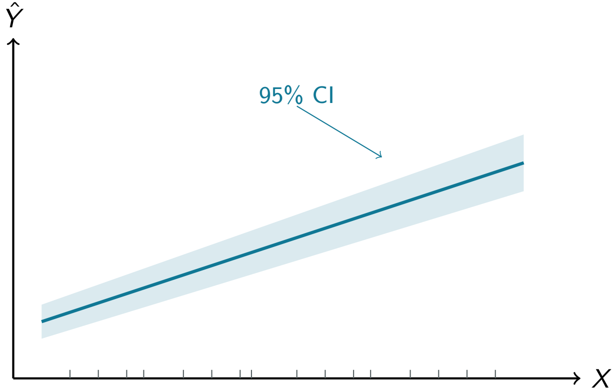
Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = "no effect"
- When are coefficient plots better than tables?
  - → Many predictors
  - → Comparing across model specifications
  - → Audience that finds tables intimidating

Coefficient plots with `modelplot()`

- Visual alternative to coefficient tables
- `modelplot(model)`
  - → Point = estimate, line = 95% CI
  - → Vertical line at zero = “no effect”
- When are coefficient plots better than tables?
  - → Many predictors
  - → Comparing across model specifications
  - → Audience that finds tables intimidating
- `modelplot(list("Model 1" = m1, "Model 2" = m2))`

Prediction plots: the gold standard



- Shows the **full relationship**, not just one number
- Any audience can read it
- `plot_predictions(model, condition = "x")`

Tables vs. plots: when to use which

	Table	Plot
Exact numbers needed	✓	
Many models side by side	✓	
Conveying one key relationship		✓
Non-specialist audience		✓
Interactions / non-linearities		✓
Appendix / robustness checks	✓	

- Best practice: prediction plots in the main text, tables in the appendix

Tables vs. plots: when to use which

	Table	Plot
Exact numbers needed	✓	
Many models side by side	✓	
Conveying one key relationship		✓
Non-specialist audience		✓
Interactions / non-linearities		✓
Appendix / robustness checks	✓	

- Best practice: prediction plots in the main text, tables in the appendix
- Both are easy to produce with `modelsummary` and `marginaleffects`

Simulation-based uncertainty

- All quantities of interest have **uncertainty**

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw
  - → Use the distribution of results as the uncertainty estimate

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw
  - → Use the distribution of results as the uncertainty estimate
- Why simulate?

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw
  - → Use the distribution of results as the uncertainty estimate
- Why simulate?
  - → Works for any quantity, no matter how complex

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw
  - → Use the distribution of results as the uncertainty estimate
- Why simulate?
  - → Works for any quantity, no matter how complex
  - → Easy to combine multiple quantities

Simulation-based uncertainty

- All quantities of interest have **uncertainty**
- Standard approach: delta method (a linear approximation for SEs of nonlinear functions)
- Alternative: **simulation**
  - → Draw $\tilde{\beta}$ from $N(\hat{\beta}, \hat{V})$ many times
  - → Compute the quantity of interest for each draw
  - → Use the distribution of results as the uncertainty estimate
- Why simulate?
  - → Works for any quantity, no matter how complex
  - → Easy to combine multiple quantities
  - → Confidence intervals are automatic

Simulation: the logic

Estimate model
$\hat{\beta}, \hat{V}$

```
m = lm(y ~ x, data = df)
```

↓

Draw $\tilde{\beta}^{(s)}$ from
$N(\hat{\beta}, \hat{V})$, $s = 1, \ldots, S$

```
mvrnorm(1000, coef(m), vcov(m))
```

↓

For each draw, compute
quantity of interest $Q^{(s)}$

```
sim_pred = sim_betas %*% x_scenario
```

↓

Use distribution of
$Q^{(1)}, \ldots, Q^{(S)}$

```
quantile(sim_pred, c(0.025, 0.975))
```

Worked example: predicted values with uncertainty

```
m = lm(income ~ age + education + female, data = df)

# Using marginaleffects (delta method)
predictions(m,
  newdata = datagrid(age = 40, education = 16, female = 1))

# Using simulation
library(MASS)
sim_b = mvrnorm(1000, coef(m), vcov(m))
x = c(1, 40, 16, 1) # intercept, age, educ, female
sim_pred = sim_b %*% x
quantile(sim_pred, c(0.025, 0.5, 0.975))
```

- Both approaches give (nearly) identical results

Roadmap

Beyond Coefficient Tables

Predicted Values and Marginal Effects

Presenting Results

Diagnostics

Wrap-up

What can go wrong?

- OLS assumes:

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)

What can go wrong?

- OLS assumes:
  - $\rightarrow$ Linear relationship between X and $E[Y]$
  - $\rightarrow$ Constant error variance (**homoskedasticity**)
  - $\rightarrow$ No extreme outliers driving results

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)
  - → No extreme outliers driving results
- When these fail:

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)
  - → No extreme outliers driving results
- When these fail:
  - → Coefficients may be biased (non-linearity)

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)
  - → No extreme outliers driving results
- When these fail:
  - → Coefficients may be biased (non-linearity)
  - → Standard errors may be wrong (heteroskedasticity)

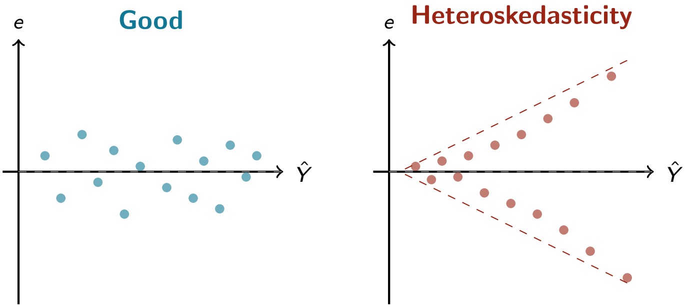
What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)
  - → No extreme outliers driving results
- When these fail:
  - → Coefficients may be biased (non-linearity)
  - → Standard errors may be wrong (heteroskedasticity)
  - → Results may be driven by a few observations (outliers)

What can go wrong?

- OLS assumes:
  - → Linear relationship between X and $E[Y]$
  - → Constant error variance (**homoskedasticity**)
  - → No extreme outliers driving results
- When these fail:
  - → Coefficients may be biased (non-linearity)
  - → Standard errors may be wrong (heteroskedasticity)
  - → Results may be driven by a few observations (outliers)
- How do we check? **Residual diagnostics**

Residuals vs. fitted values



- In R: `plot(model, which = 1)`
- Look for patterns: funnel shape, curves, clusters

You plot residuals vs. fitted values and see a clear funnel shape.

What does this tell you about your model?

What would you do about it?

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - → Coefficients are still unbiased

Heteroskedasticity

- **Heteroskedasticity**: $\mathrm{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - $\rightarrow$ Coefficients are still unbiased
  - $\rightarrow$ But standard errors are **wrong**

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - $\rightarrow$ Coefficients are still unbiased
  - $\rightarrow$ But standard errors are **wrong**
  - $\rightarrow$ So confidence intervals and p-values are unreliable

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - → Coefficients are still unbiased
  - → But standard errors are **wrong**
  - → So confidence intervals and p-values are unreliable
- Very common in practice:

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - $\rightarrow$ Coefficients are still unbiased
  - $\rightarrow$ But standard errors are **wrong**
  - $\rightarrow$ So confidence intervals and p-values are unreliable
- Very common in practice:
  - $\rightarrow$ Income models (richer countries, more variance)

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - → Coefficients are still unbiased
  - → But standard errors are **wrong**
  - → So confidence intervals and p-values are unreliable
- Very common in practice:
  - → Income models (richer countries, more variance)
  - → Any outcome with natural bounds

Heteroskedasticity

- **Heteroskedasticity**: $\text{Var}(\varepsilon \mid X)$ is not constant
- Consequences:
  - $\rightarrow$ Coefficients are still unbiased
  - $\rightarrow$ But standard errors are **wrong**
  - $\rightarrow$ So confidence intervals and p-values are unreliable
- Very common in practice:
  - $\rightarrow$ Income models (richer countries, more variance)
  - $\rightarrow$ Any outcome with natural bounds
  - $\rightarrow$ Binary outcomes (always heteroskedastic, as in LPM)

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called “robust” or “White” standard errors

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`
  - → `modelsummary(model, vcov = "HC3")`

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`
  - → `modelsummary(model, vcov = "HC3")`
  - → `estimatr::lm_robust(y ~ x, data = df)`

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`
  - → `modelsummary(model, vcov = "HC3")`
  - → `estimatr::lm_robust(y ~ x, data = df)`
- Best practice: **always use robust SEs**

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`
  - → `modelsummary(model, vcov = "HC3")`
  - → `estimatr::lm_robust(y ~ x, data = df)`
- Best practice: **always use robust SEs**
  - → If homoskedastic: robust SEs $\approx$ classical SEs

Robust standard errors

- The fix: **heteroskedasticity-consistent (HC) standard errors**
- Also called "robust" or "White" standard errors
- In R (multiple options):
  - → `modelsummary(model, vcov = "robust")`
  - → `modelsummary(model, vcov = "HC3")`
  - → `estimatr::lm_robust(y ~ x, data = df)`
- Best practice: **always use robust SEs**
  - → If homoskedastic: robust SEs $\approx$ classical SEs
  - → If heteroskedastic: robust SEs are correct, classical are not

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:
  → Income, GDP, population, firm size, casualties

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:
  $\rightarrow$ Income, GDP, population, firm size, casualties
- Taking $\log(X)$ compresses the right tail

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:
  - → Income, GDP, population, firm size, casualties
- Taking $\log(X)$ compresses the right tail
- Makes the distribution more symmetric

Log transformations: when and why

- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:
  - → Income, GDP, population, firm size, casualties
- Taking $\log(X)$ compresses the right tail
- Makes the distribution more symmetric
- Can reduce heteroskedasticity

Log transformations: when and why

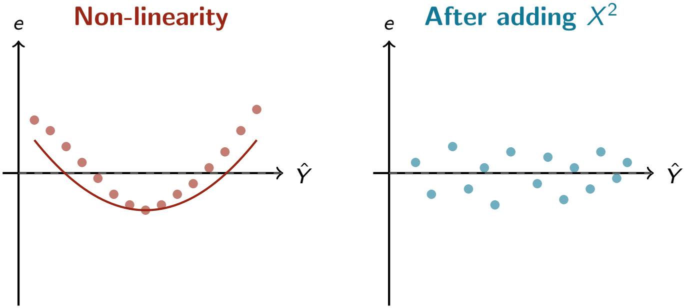
- Another common fix for heteroskedasticity and non-linearity: **transform the variables**
- Many variables are right-skewed:
  → Income, GDP, population, firm size, casualties
- Taking $\log(X)$ compresses the right tail
- Makes the distribution more symmetric
- Can reduce heteroskedasticity
- But interpretation changes!

Interpreting logs

Model	**Equation**	**Interpretation of** β_1
Level-level	$Y = \beta_0 + \beta_1 X$	$\Delta X = 1 \Rightarrow \Delta Y = \beta_1$
Log-level	$\log(Y) = \beta_0 + \beta_1 X$	$\Delta X = 1 \Rightarrow \%\Delta Y \approx 100 \cdot \beta_1$
Level-log	$Y = \beta_0 + \beta_1 \log(X)$	$\%\Delta X = 1 \Rightarrow \Delta Y \approx \beta_1/100$
Log-log	$\log(Y) = \beta_0 + \beta_1 \log(X)$	$\%\Delta X = 1 \Rightarrow \%\Delta Y \approx \beta_1$

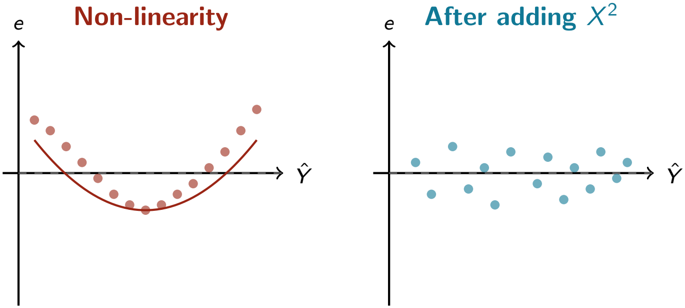
- **Just use plots** of predicted values

Residual plots: checking linearity



- A U-shape in residuals ⇒ missing non-linearity

Residual plots: checking linearity



- A U-shape in residuals ⇒ missing non-linearity
- Fixes: add X^2, use $\log(X)$, or use a more flexible model

Influential observations

- Some observations can disproportionately influence results

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation i

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation i
  - → In R: `plot(model, which = 4)`

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation i
  - $\rightarrow$ In R: `plot(model, which = 4)`
  - $\rightarrow$ Observations with Cook's $D > 4/n$: investigate further

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation i
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation i
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)
- What to do with influential observations?

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation *i*
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)
- What to do with influential observations?
  - → Do NOT automatically delete them

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation *i*
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)
- What to do with influential observations?
  - → Do NOT automatically delete them
  - → Check if they are data errors

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation *i*
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)
- What to do with influential observations?
  - → Do NOT automatically delete them
  - → Check if they are data errors
  - → Re-run the model without them as a robustness check

Influential observations

- Some observations can disproportionately influence results
- **Cook's distance**: measures how much the model changes if you remove observation *i*
  - → In R: `plot(model, which = 4)`
  - → Observations with Cook's $D > 4/n$: investigate further
- **Alternatively:** just loop through models dropping one observation (or group of) each time (*leave-one-out*)
- What to do with influential observations?
  - → Do NOT automatically delete them
  - → Check if they are data errors
  - → Re-run the model without them as a robustness check
  - → Report both results

Roadmap

Beyond Coefficient Tables

Predicted Values and Marginal Effects

Presenting Results

Diagnostics

Wrap-up

Summary: key takeaways

- Raw coefficients are rarely enough — compute **quantities of interest**

Summary: key takeaways

- Raw coefficients are rarely enough — compute **quantities of interest**
- Predicted values, marginal effects, and first differences (all via `marginaleffects`)

Summary: key takeaways

- Raw coefficients are rarely enough — compute **quantities of interest**
- Predicted values, marginal effects, and first differences (all via `marginaleffects`)
- Present with plots (main text) and tables (appendix)

Summary: key takeaways

- Raw coefficients are rarely enough — compute **quantities of interest**
- Predicted values, marginal effects, and first differences (all via `marginaleffects`)
- Present with plots (main text) and tables (appendix)
- Always use robust standard errors and check residual diagnostics

Summary: key takeaways

- Raw coefficients are rarely enough — compute **quantities of interest**
- Predicted values, marginal effects, and first differences (all via `marginaleffects`)
- Present with plots (main text) and tables (appendix)
- Always use robust standard errors and check residual diagnostics
- Log-transform skewed variables (but interpret carefully)

For next week

- Complete Assignment 4
- Next session: Best Practices in Computing
  → Reproducible workflows
  → Project organization
  → Writing clean R code

Questions?