Modeling Nonlinear Moderation Effects in R

From Multiple Regression to Structural Equation Modeling

Tuo Liu   Philipp Franikowski   Andrea Hildebrandt

Introduction and Welcome

09:00–09:15

Tuo Liu

- since 2022: Research Assistant, Department of Educational Psychology with emphasis on Consultation, Diagnostics and Evaluation, Goethe-University Frankfurt, Germany
- 2019–2022: Research Assistant, Division for Psychological Methods and Statistics, Department of Psychology, Carl von Ossietzky University Oldenburg, Germany
- 2016–2018: M.Sc. in Media & Instructional Psychology, Chemnitz University of Technology, Chemnitz, Germany
- 2011–2015: B.A. in Communication Science, University of International Relations, Beijing, PR China

@LiuThorsten | t.liu@psych.uni-frankfurt.de

Philipp Franikowski

- since 2022: Postdoc, Statistical Methods at Institute for Educational Quality Improvement, HU Berlin
- 2017–2022: Predoc at University of Greifswald
  - Dissertation: *Semantic or Affective Primacy? Perceptual Latencies of Object Recognition and Affect Measured With Temporal Judgments and Speeded Reaction Time Tasks*
- 2012–2017: Diploma in Psychology at University of Greifswald

@franikowsp | philipp.franikowski@iqb.hu-berlin.de | franikowsp |

Prof. Andrea Hildebrandt

- since 2018: Full Professor for Psychological Methods and Statistics, Carl von Ossietzky University Oldenburg
- 2013–2018: Assistant Professor for Psychological Assessment and Personality Psychology at the University of Greifswald
- 2010–2013: Postdoc at the HU Berlin, Department of Psychology, Psychological Methods
- 2010–2011: Postdoc at the Universität Duisburg-Essen, Department of Psychology, Educational and Psychological Assessment
- 2007–2010: Predoc at the at Institute for Educational Quality Improvement, HU Berlin
- 2010: Dr. rer. nat. in Psychology at the HU Berlin
    - Dissertation: *Individual and Age-Related Differences in Face Cognition*

@ahildepsych | andrea.hildebrandt@uol.de

Who are you?

1 sentence each:

- Where are you from?
- What is your research interest?
- What are your expectations regarding the workshop?
- What is your experience with R and SEM?

HTML Slides

https://franikowsp.github.io/lsem-mnsem/

PDF Slides and Material

https://github.com/franikowsp/lsem-mnsem-materials

Overview

- 09:00–09:15 | **Introduction** and Welcome
- 09:15–09:45 | **Recap 1** Moderation and Regression
- 09:45–11:15 | **Session 1** Moderated Nonlinear Regression
- 11:15–11:45 | ☕ Coffee Break
- 11:45–13:00 | **Session 2** Local Regression
- 13:00–14:00 | 🛎 Lunch Break
- 14:00–14:30 | **Recap 2** Structural Equation Models and Measurement Invariance
- 14:30–15:30 | **Session 3** Moderated Nonlinear SEM
- 15:30–16:00 | ☕ Coffee Break
- 16:00–17:00 | **Session 4** Local SEM
- 17:00–17:45 | **Session 5** Advanced Topics and Recommendations
- 17:45–18:00 | **Questions** and Feedback

Recap 1

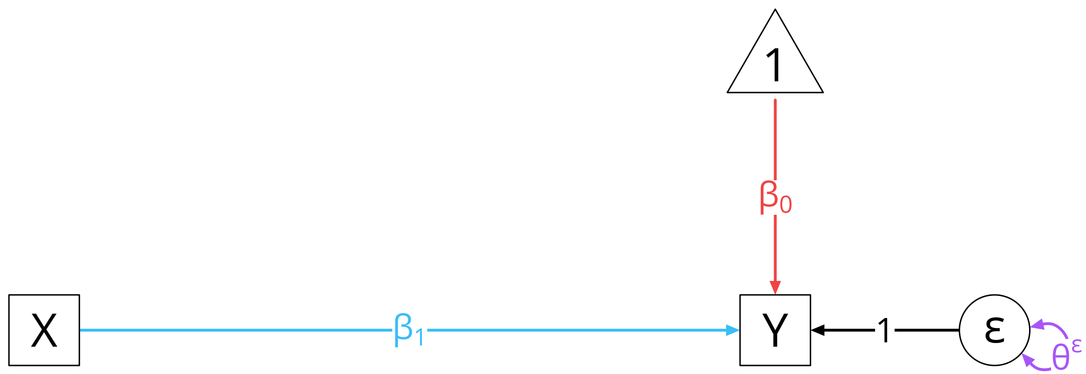
Moderation and Regression

09:15–09:45

Simple Regression

- Linear regression with 1 predictor:

$$Y = \beta_0 + \beta_1 x + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$



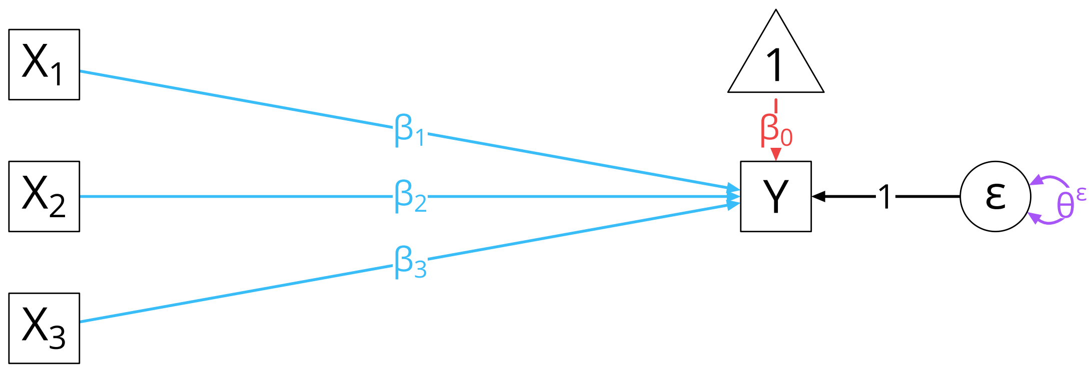
- **Coefficients:**
  - β_0 = intercept
  - β_1 = slope, i.e., linear effect of X
  - θ_ε = residual variance

Simple Regression

Multiple Regression

- Linear regression with p predictors:

$$Y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

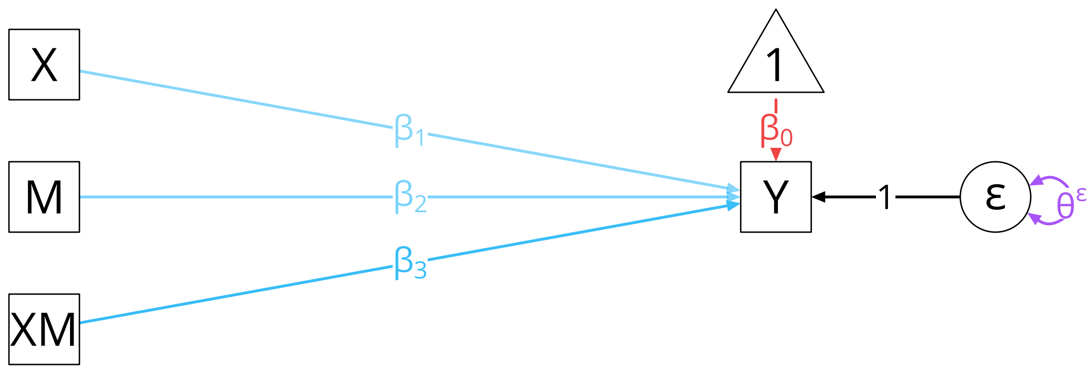


- **Coefficients:**
  - β_0 = intercept
  - β_j = slope, i.e., linear effect of X_j
  - θ_ε = residual variance

Moderated Regression

- Moderated regression with 2 predictors, the usual definition:

$$Y = \beta_0 + \beta_1 X + \beta_2 M + \beta_3 XM + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$



Moderated Regression

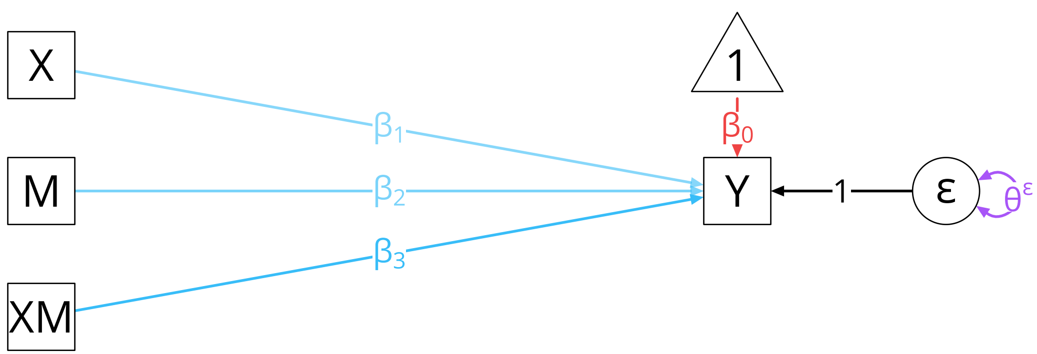
- **Coefficients:**
  - β_0 = intercept
  - β_1 = linear effect of X
  - β_2 = linear effect of M
  - β_3 = interaction effect of the product term XM

Moderation

- **Question:** Does the relationship between an independent variable X and a dependent variable Y change as a function of a third variable, known as the moderator M?

Moderator Variables

- Third variables that influence the **strength** or **direction** of the relationship between an independent variable X and a dependent variable Y
- Help to explain when or under what conditions an X affects Y



Moderated Regression

Moderation

- **Examples** in Psychology:
  - **Stress and Performance**: Social support (M) might moderate the negative impact of stress (X) on performance (Y)
    - E.g., weaker impacts with high social support
  - **Therapy Effectiveness**: Severity of symptoms (M) could moderate the effectiveness (Y) of a particular therapy (X)
    - E.g., greater improvements for those with more severe symptoms
  - **Work Satisfaction**: Job autonomy (M) might moderate the relationship between workload (X) and job satisfaction (Y)
    - E.g., high autonomy mitigates the negative effects of a heavy workload

Interaction Terms

$$Y = \beta_0 + \beta_1 X + \beta_2 M + \beta_3 XM + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

- **Assessment**: adding the multiplication term $X \times M$
- **Interpretation**: How does the relationship between the X and Y change at different levels of the M?
  - **Significance of interaction term**: significant interaction term indicates that the effect of X on the Y depends on the level of M
  - **Simple slopes**: relationship between the X and Y at specific values of M (e.g., $\bar{M} \pm 1SD$).
  - **Plotting interactions**: illustration of how the relationship changes across different levels of the moderator

Dataset 1

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

- **Model:** impact of study hours on academic performance, moderated by sleep quality
- **Variables:**
  - Y = `aca_per`: academic performance
  - X = `stu_hou`: study hours per week (range = 0–20)
  - M = `sle_qua`: sleep quality (range = 0–10)

Example: No Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
fit <- lm(aca_per ~ stu_hou * sle_qua, data = performance)
summary(fit)$coefficients
```

```
#>                   Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)     63.1901876 2.21158691  28.572328 7.200758e-107
#> stu_hou         -2.4506362 0.19104897 -12.827267  9.951223e-33
#> sle_qua         -2.7216562 0.37323163  -7.292137  1.214314e-12
#> stu_hou:sle_qua  0.3354037 0.03238114  10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `(Intercept)`: β_0 = estimated value of Y when both X and M = 0, i.e., the baseline `aca_per` is 63.19
    - **Note:** `stu_hou` and `sle_qua` are rarely 0 in real life → intercept may not have a practical interpretation

Example: No Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```
fit <- lm(aca_per ~ stu_hou * sle_qua, data = performance)
summary(fit)$coefficients
```

```
#>                   Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)     63.1901876 2.21158691  28.572328 7.200758e-107
#> stu_hou         -2.4506362 0.19104897 -12.827267  9.951223e-33
#> sle_qua         -2.7216562 0.37323163  -7.292137  1.214314e-12
#> stu_hou:sle_qua  0.3354037 0.03238114  10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `stu_hou`: β_1 = main effect of X on Y when M = 0, i.e., if `stu_hou` increases by 1 unit, the `aca_per` decreases by 2.45 units, holding `sle_qua` constant
    - **Note:** negative main effect would suggest that studying less leads to better `aca_per`, assuming a `sle_qua` of 0

Example: No Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
fit <- lm(aca_per ~ stu_hou * sle_qua, data = performance)
summary(fit)$coefficients
```

```
#>                  Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)    63.1901876 2.21158691  28.572328 7.200758e-107
#> stu_hou        -2.4506362 0.19104897 -12.827267  9.951223e-33
#> sle_qua        -2.7216562 0.37323163  -7.292137  1.214314e-12
#> stu_hou:sle_qua  0.3354037 0.03238114  10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `sle_qua`: β_2 = main effect of M on Y when $X = 0$, i.e., if `sle_qua` increases by 1 unit, the `aca_per` decreases by 2.72 units, holding `stu_hou` constant
    - **Note:** negative main effect would suggest that having a lower sleep quality leads to a better `aca_per`, assuming 0 `stu_hou`

Example: No Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
fit <- lm(aca_per ~ stu_hou * sle_qua, data = performance)
summary(fit)$coefficients
```

```
#>                   Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)     63.1901876 2.21158691  28.572328 7.200758e-107
#> stu_hou         -2.4506362 0.19104897 -12.827267  9.951223e-33
#> sle_qua         -2.7216562 0.37323163  -7.292137  1.214314e-12
#> stu_hou:sle_qua  0.3354037 0.03238114  10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `stu_hou:sle_qua`: β_3 = interaction effect / moderation of the effect of X on Y by M
    - **Note:** combined effect of `stu_hou` and `sle_qua` on `aca_per` is significantly positive, i.e., higher sleep quality enhances the positive effect of study time on exam performance.

Centering

- Preprocessing step of subtracting the mean of a variable from each value of that variable, i.e., new mean = 0
- **Consequences** for moderated regression:
  - **Reduced multicollinearity**: centering variables before creating the interaction term reduces the risk of multicollinearity (i.e., high correlation of independent variables in a regression model)
    - **Problem:** potentially difficult to assess individual contributions of each variable
  - **Better interpretability of main effects**: more straightforward interpretation of the main effects, i.e., The coefficients of the main effects β_1 and β_2 represent the effect of each variable when the other variable is at its mean
    - **Problem:** without centering, coefficients potentially difficult to interpret because they include the interaction effect
  - **Improved numerical stability**: improves numerical stability of regression estimates, particularly when dealing with large datasets or variables with large values.

Example: Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
performance$stu_hou_c <- performance$stu_hou - mean(performance$stu_hou)
performance$sle_qua_c <- performance$sle_qua - mean(performance$sle_qua)
fit <- lm(aca_per ~ stu_hou_c * sle_qua_c, data = performance)
summary(fit)$coefficients
```

```
#>                      Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept)        41.9143292 0.54314140 77.170198 1.816486e-278
#> stu_hou_c          -0.7760596 0.09565223 -8.113346  3.902744e-15
#> sle_qua_c           0.6007436 0.18698805  3.212738  1.400240e-03
#> stu_hou_c:sle_qua_c  0.3354037 0.03238114 10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `(Intercept)`: β_0 = estimated value of Y when both X_C and M_C = 0 or $X = \bar{X}$ and $M = \bar{M}$, i.e., the average `aca_per` for a student with average `stu_hou` and `sle_qua` is 41.91

Example: Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
performance$stu_hou_c <- performance$stu_hou - mean(performance$stu_hou)
performance$sle_qua_c <- performance$sle_qua - mean(performance$sle_qua)
fit <- lm(aca_per ~ stu_hou_c * sle_qua_c, data = performance)
summary(fit)$coefficients
```

```
#>                       Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept)         41.9143292 0.54314140 77.170198 1.816486e-278
#> stu_hou_c           -0.7760596 0.09565223 -8.113346  3.902744e-15
#> sle_qua_c            0.6007436 0.18698805  3.212738  1.400240e-03
#> stu_hou_c:sle_qua_c  0.3354037 0.03238114 10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `stu_hou_c`: β_1 = main effect of X on Y when $M = \bar{M}$, i.e., if `stu_hou_c` increases by 1 unit, the `aca_per` decreases by 0.78 units on average `sle_qua`
    - **Note:** negative main effect would suggest that studying less than the average amount leads to better `aca_per`, assuming average `sle_qua`

Example: Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```
performance$stu_hou_c <- performance$stu_hou - mean(performance$stu_hou)
performance$sle_qua_c <- performance$sle_qua - mean(performance$sle_qua)
fit <- lm(aca_per ~ stu_hou_c * sle_qua_c, data = performance)
summary(fit)$coefficients
```

```
#>                       Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)         41.9143292 0.54314140  77.170198 1.816486e-278
#> stu_hou_c           -0.7760596 0.09565223  -8.113346  3.902744e-15
#> sle_qua_c            0.6007436 0.18698805   3.212738  1.400240e-03
#> stu_hou_c:sle_qua_c  0.3354037 0.03238114  10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `sle_qua_c`: β_2 = main effect of M on Y when $X = \bar{X}$, i.e., if `sle_qua_c` increases by 1 unit, the `aca_per` increases by 0.60 units on average `stu_hou`
    - **Note:** positive main effect would suggest that having better sleep quality than the average leads to better `aca_per`, assuming average `stu_hou`

Example: Centering

Example 1.1: Academic Performance (`performance`)

We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
performance$stu_hou_c <- performance$stu_hou - mean(performance$stu_hou)
performance$sle_qua_c <- performance$sle_qua - mean(performance$sle_qua)
fit <- lm(aca_per ~ stu_hou_c * sle_qua_c, data = performance)
summary(fit)$coefficients
```

```
#>                       Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept)         41.9143292 0.54314140 77.170198 1.816486e-278
#> stu_hou_c           -0.7760596 0.09565223 -8.113346  3.902744e-15
#> sle_qua_c            0.6007436 0.18698805  3.212738  1.400240e-03
#> stu_hou_c:sle_qua_c  0.3354037 0.03238114 10.357996  6.828773e-23
```

- **Coefficient interpretation:**
  - `stu_hou:sle_qua`: β_3 = interaction effect / moderation of the effect of X on Y by M
    - **Note:** combined effect of `stu_hou` and `sle_qua` on `aca_per` is significantly positive, i.e., higher sleep quality enhances the positive effect of study time on exam performance.

Comparison: Centering

Uncentered Variables

- **Intercept:** Y when both X and M are 0 (may not be practical)
- **Main effects:** effect of one variable when the other variable is 0 (potentially misleading interpretations)
- **Interaction Effect:** change in the relationship between the independent variables (interpretation more complex due to the uncentered nature)
- **Note:** no centering might required for variables with a natural 0 point (e.g., age)

Centered Variables

- **Intercept:** Y for average X and average M (more meaningful)
- **Main effects:** effect of each variable at the average level of the other variable (clearer and more interpretable insights)
- **Interaction effect:** change in the relationship between the independent variables (clearer intepretation: how deviations from the average in both variables interact to influence exam performance)

Centering: Model Fit

Centering does not affect model fit, but only the interpretation of the coefficients.

Uncentered Variables

```
fit <- lm(aca_per ~ stu_hou * sle_qua,
          data = performance)
fit_s <- summary(fit)
fit_s[c("r.squared", "adj.r.squared", "fstatistic"
```

```
#> $r.squared
#> [1] 0.2657669
#>
#> $adj.r.squared
#> [1] 0.2613259
#>
#> $fstatistic
#>     value     numdf     dendf
#>  59.84492   3.00000 496.00000
```

Centered Variables

```
fit <- lm(aca_per ~ sle_qua_c * stu_hou_c,
          data = performance)
fit_s <- summary(fit)
fit_s[c("r.squared", "adj.r.squared", "fstatistic"
```

```
#> $r.squared
#> [1] 0.2657669
#>
#> $adj.r.squared
#> [1] 0.2613259
#>
#> $fstatistic
#>     value     numdf     dendf
#>  59.84492   3.00000 496.00000
```

Example: Simple Slopes

Example 1.1: Academic Performance (`performance`)

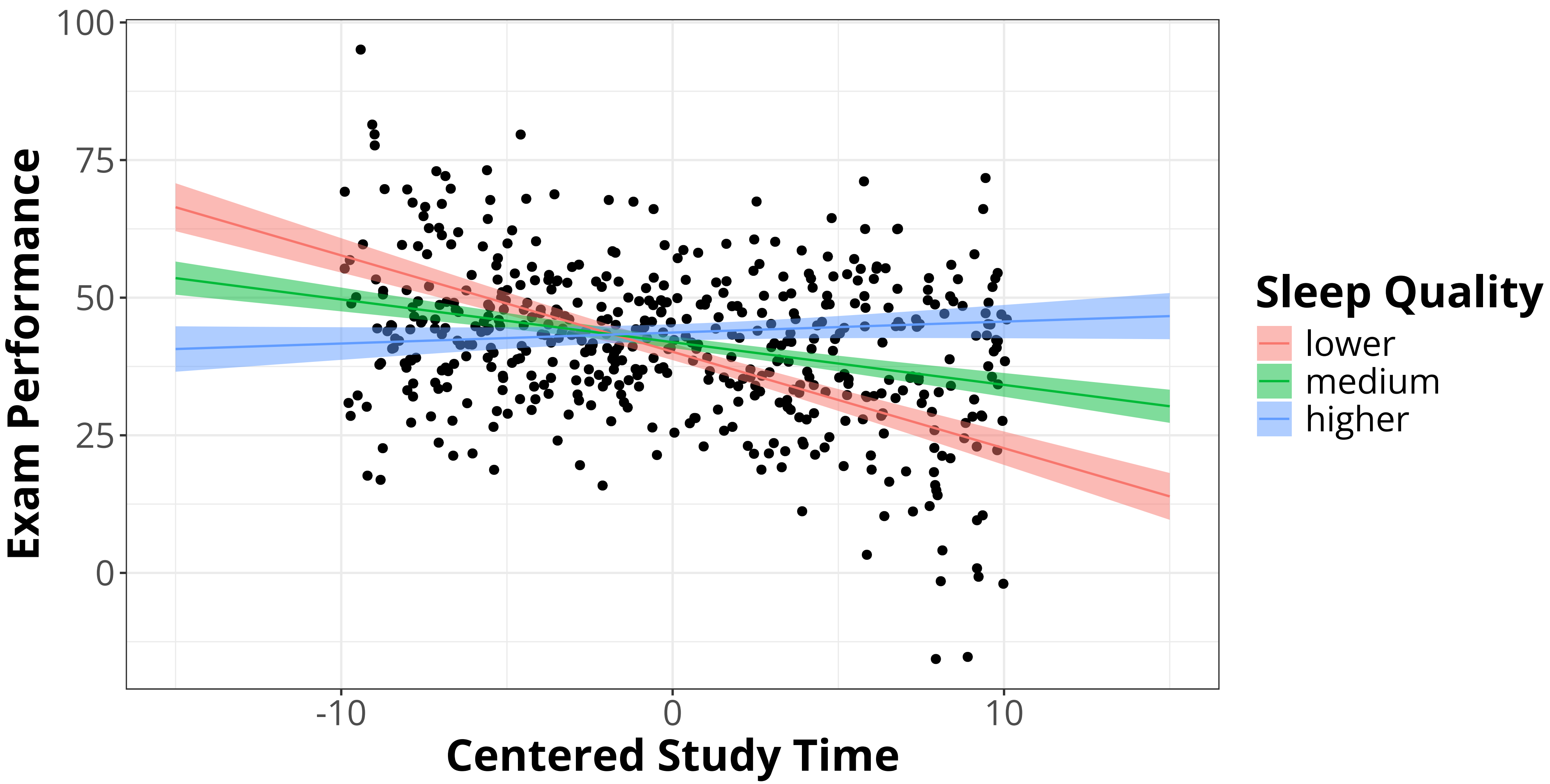
We are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship.

```r
m_sq <- mean(performance$sle_qua_c)
sd_sq <- sd(performance$sle_qua_c)
focal_points <- c(m_sq - sd_sq, m_sq, m_sq + sd_sq)
emtrends(fit, specs = ~ sle_qua_c | stu_hou_c, var = "stu_hou_c",
         at = list(sle_qua_c = focal_points))
```

```
#> stu_hou_c = 6.4e-16:
#>  sle_qua_c stu_hou_c.trend     SE  df lower.CL upper.CL
#>      -2.91          -1.751 0.1368 496  -2.0202   -1.483
#>       0.00          -0.776 0.0957 496  -0.9640   -0.588
#>       2.91           0.199 0.1316 496  -0.0592    0.458
#>
#> Confidence level used: 0.95
```

- Lower `sle_qua`: increased `stu_hou` strongly decreases `aca_per` significantly
- Average `sle_qua`: increased `stu_hou` decreases `aca_per`
- High `sle_qua`: increased `stu_hou` enhances `aca_per`

Example: Ploting Interactions

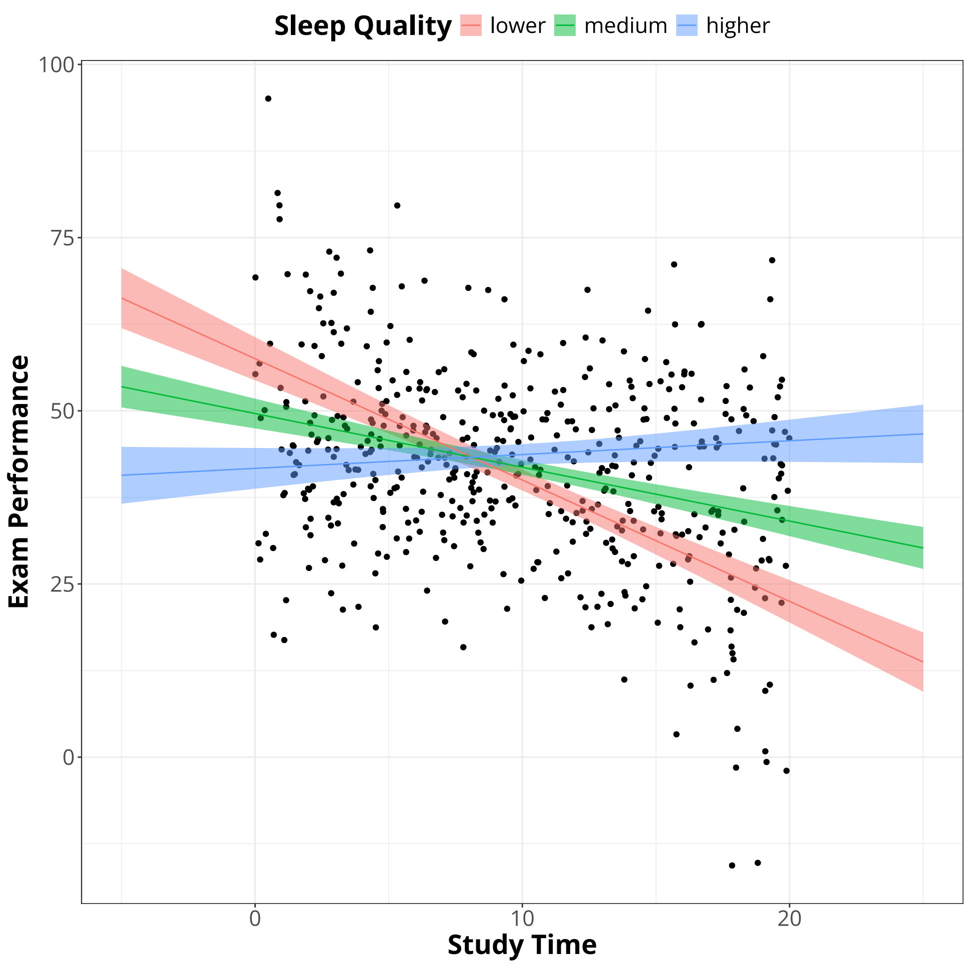


Simple slopes for average sleep quality ± 1 *SD*

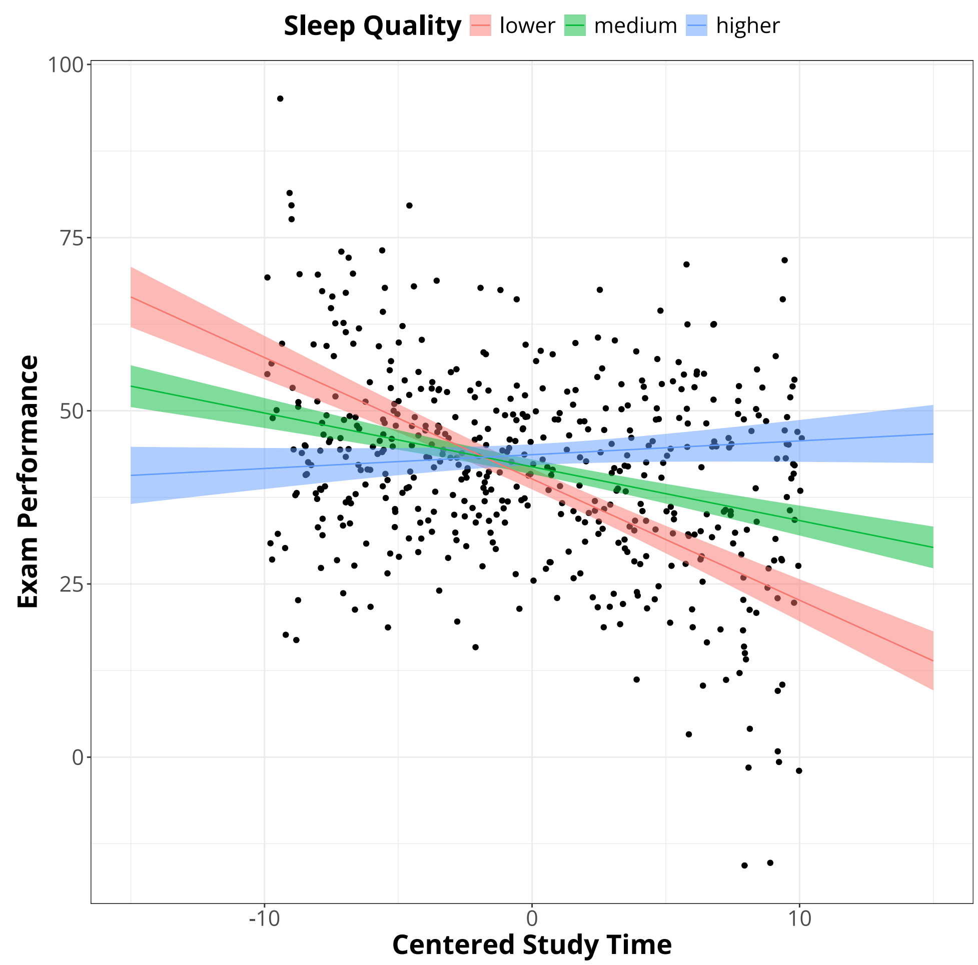
Centering: Model Fit

Centering only influences the interpretation of the coefficients.

Uncentered Variables



Centered Variables



Session 1

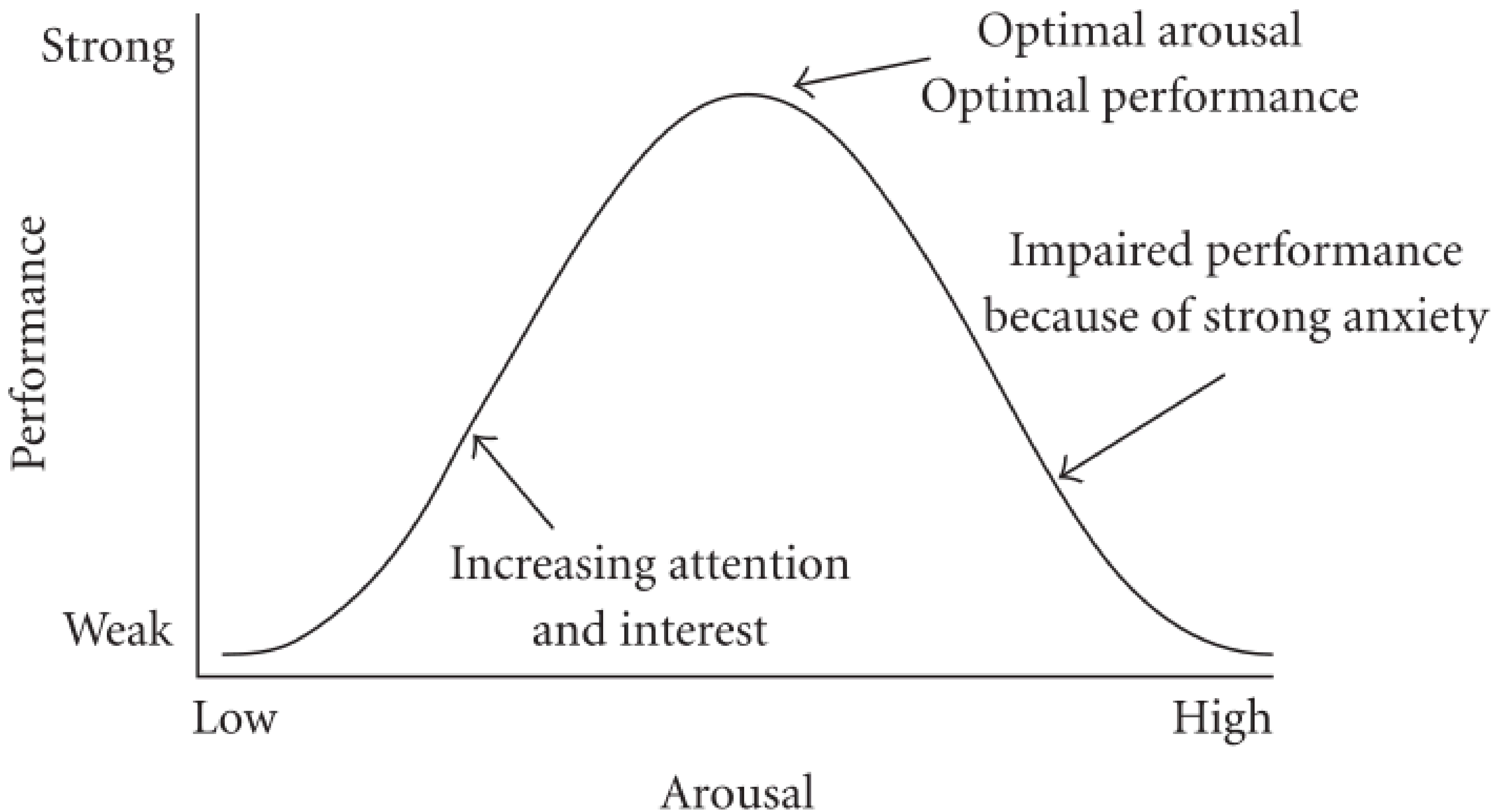
Moderated Nonlinear Regression

09:45–11:15

Why Nonlinear Relations

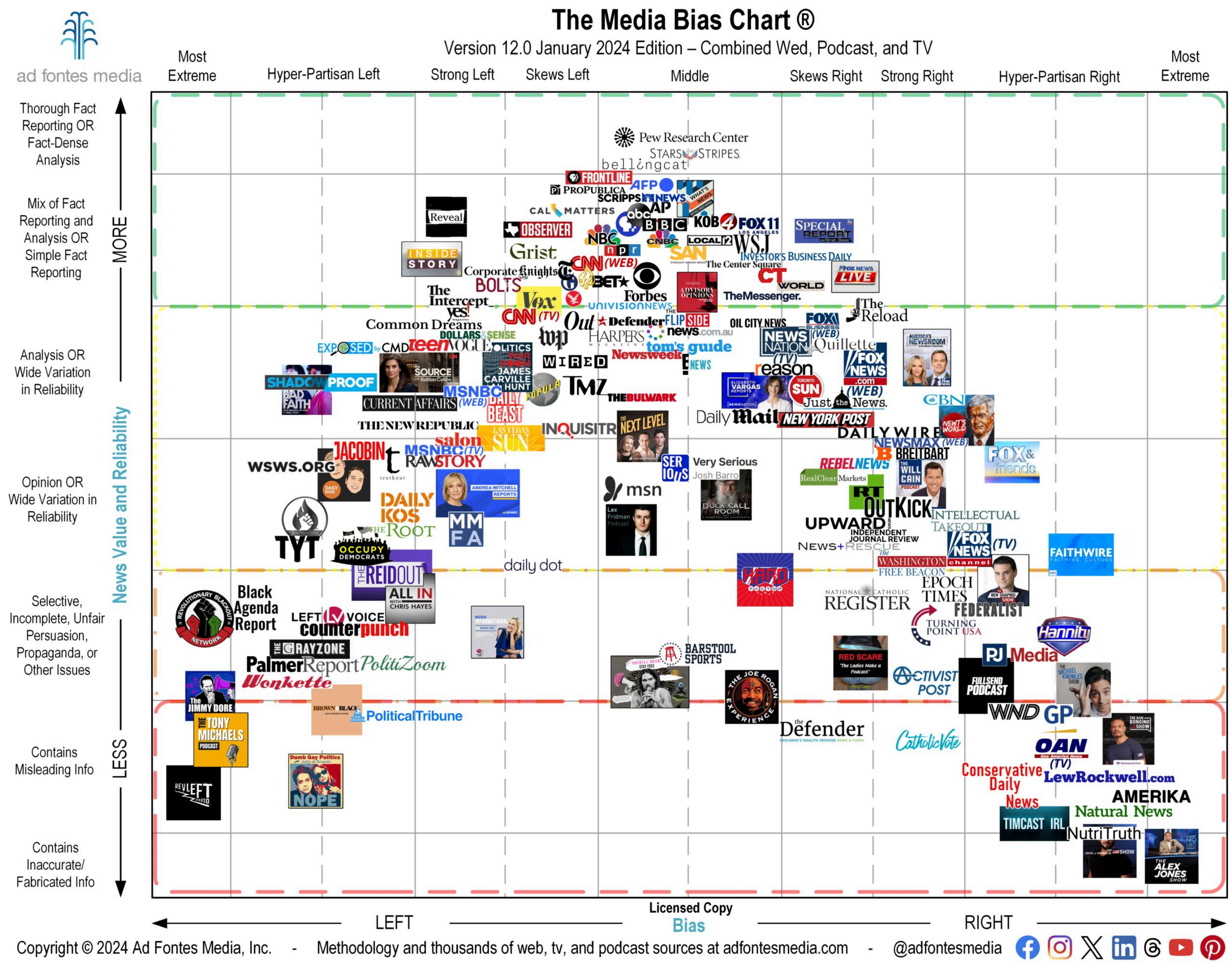
- **Linear models:** relatively easy to describe, to implement, and to interpret
  - **Limitations:** predictive power
    - Linearity assumption is almost always an approximation, and sometimes a poor one
- **Examples** from Psychology:
  - Cognitive Psychology: Yerkes–Dodson law
  - Social and Media Psychology: Media bias chart
  - Developmental Psychology: Intellectual development

Example: Nonlinearity



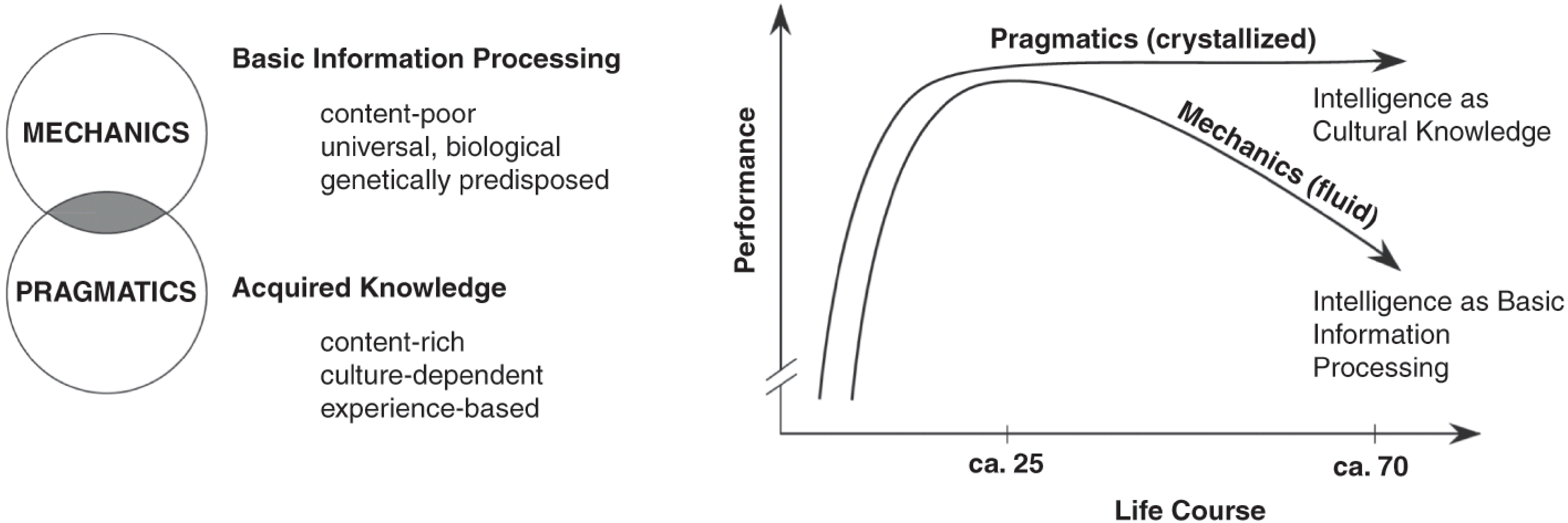
Yerkes-Dodson Law

Example: Nonlinearity



Ad Fontes Media Bias Chart

Example: Nonlinearity



Baltes' Conception of the Two curves of Intellectual Development (Hertzog, 2011)

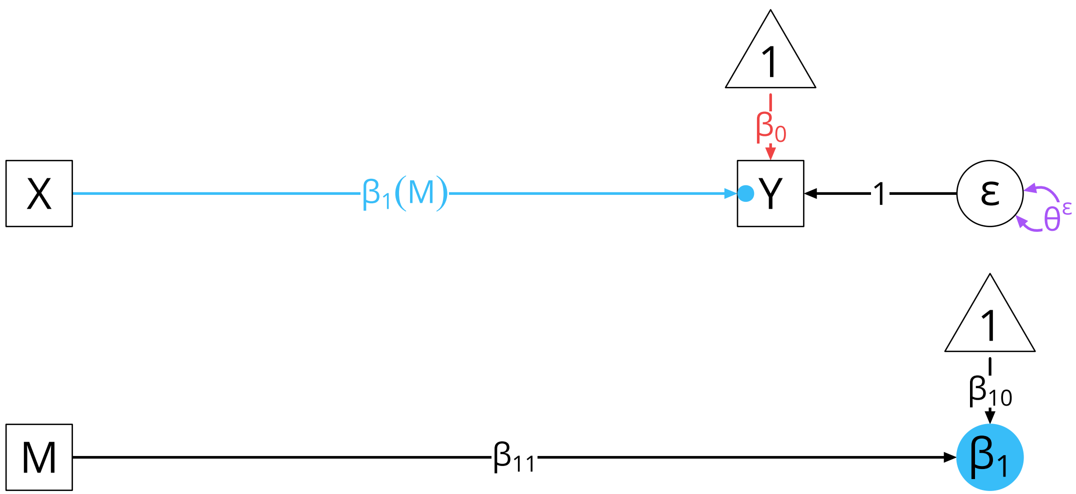
Modeling Nonlinearity

- An alternative framework for modeling interaction effects: **varying coefficient models**
  - **Fixed coefficient model:** $Y = \beta_0 + \beta_1 X + \varepsilon$
    - All coefficients are constants, i.e., β_0 and β_1
  - **Varying coefficient model:** $Y = \beta_0(Z) + \beta_1(Z) X + \varepsilon$
    - One or more coefficients vary as a function of one, i.e., $\beta_0(Z)$ and $\beta_1(Z)$, or more variables, i.e., $\beta_0(Z_1, Z_2, \ldots)$ and $\beta_1(Z_1, Z_2, \ldots)$
    - **Advantage:** captures (complex) interactions and non-linear relationships more naturally and flexibly

Varying Slope Model

- Moderated linear regression with a varying slope:

$$Y = \beta_0 + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_1(M) = \beta_{10} + \beta_{11}M$$



Varying Slope Model

- **Coefficients for β_1 on M:**
  - β_{10} = intercept
  - β_{11} = linear effect

Varying Slope Model

- Moderated linear regression with a varying slope:

$$Y = \beta_0 + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_1(M) = \beta_{10} + \beta_{11}M$$

```r
fit <- lm(aca_per ~ 1 + stu_hou_c + sle_qua_c : stu_hou_c, data = performance)
summary(fit)$coefficients
```

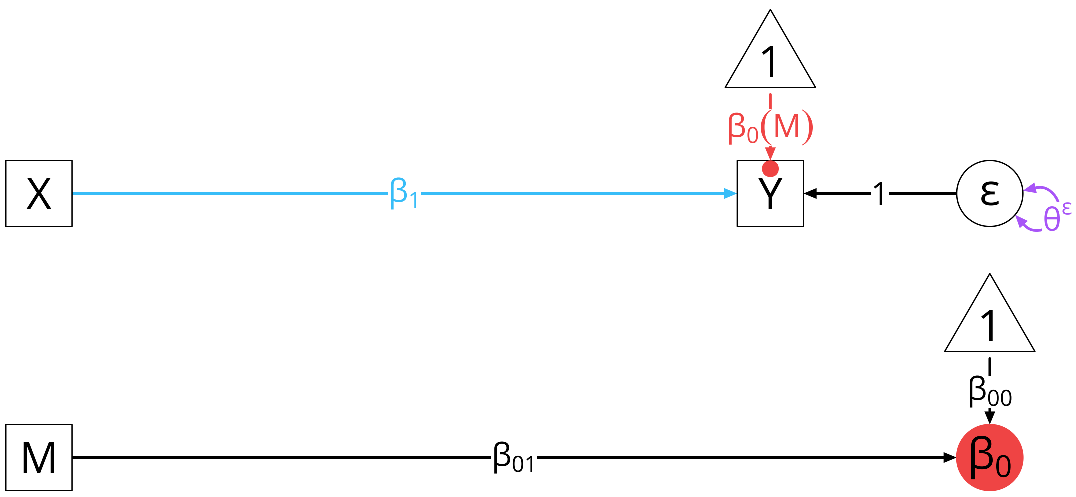
```
#>                        Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept)          41.9149537 0.54821126 76.457667 5.614952e-277
#> stu_hou_c            -0.7854273 0.09650022 -8.139124  3.222703e-15
#> stu_hou_c:sle_qua_c   0.3366632 0.03268100 10.301496  1.098879e-22
```

- **Coefficients for β_1 on M:**
  - β_{10} = -0.79, intercept
  - β_{11} = 0.34, linear effect

Varying Intercept Model

- Moderated linear regression with a varying intercept:

$$Y = \beta_0(M) + \beta_1 X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_0(M) = \beta_{00} + \beta_{01} M$$



Varying Intercept Model

- **Coefficients for β_1 on M:**
  - β_{00} = intercept
  - β_{01} = linear effect

Varying Intercept Model

- Moderated linear regression with a varying intercept:

$$Y = \beta_0(M) + \beta_1 X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_0(M) = \beta_{00} + \beta_{01} M$$

```r
fit <- lm(aca_per ~ 1 + sle_qua_c + stu_hou_c, data = performance)
summary(fit)$coefficients
```

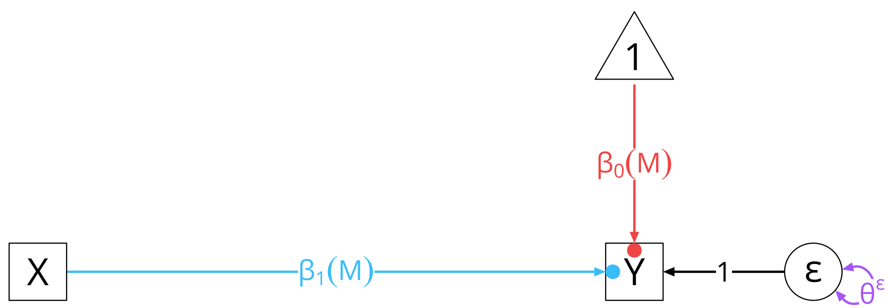
```
#>                Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept) 41.7480376  0.5981461 69.795719 5.646921e-259
#> sle_qua_c    0.6241921  0.2059995  3.030066  2.572371e-03
#> stu_hou_c   -0.7371958  0.1053040 -7.000643  8.298477e-12
```

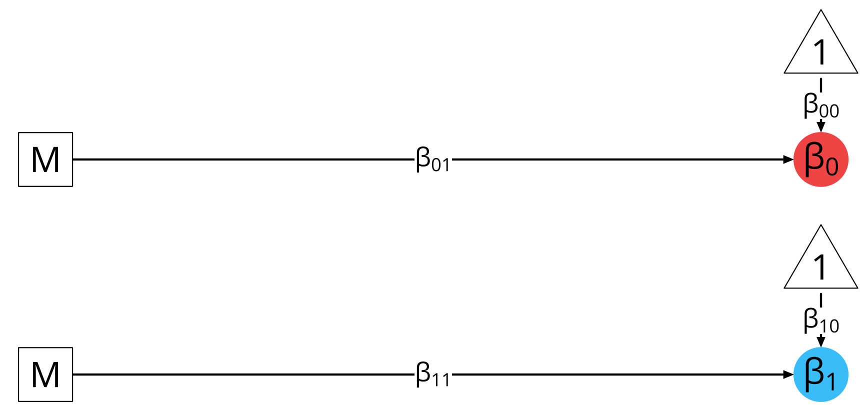
- **Coefficients for β_0 on M:**
  - β_{00} = 41.75, intercept
  - β_{01} = 0.62, linear effect

Varying Coefficient Model

- Moderated linear regression with a varying coefficients:

$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_0(M) = \beta_{00} + \beta_{01}M$$
$$\beta_1(M) = \beta_{10} + \beta_{11}M$$





Varying Coefficient Model

- Moderated linear regression with a varying coefficients:

$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

$$\beta_0(M) = \beta_{00} + \beta_{01}M; \beta_1(M) = \beta_{10} + \beta_{11}M$$

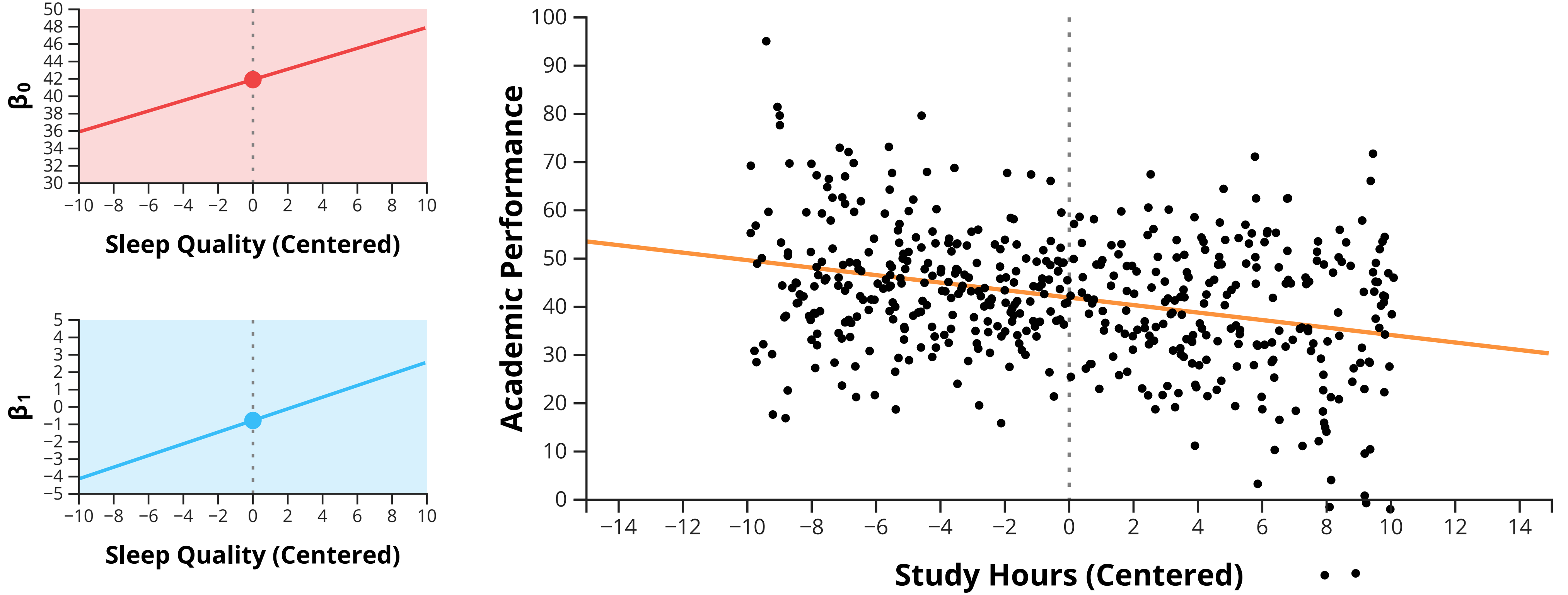
$$\hat{Y} = \beta_{00} + \beta_{01}M + (\beta_{10} + \beta_{11}M)X = \beta_{00} + \beta_{01}M + \beta_{10}X + \beta_{11}MX$$

```r
fit <- lm(aca_per ~ 1 + sle_qua_c + stu_hou_c + stu_hou_c : sle_qua_c, data = performance)
summary(fit)$coefficients
```

```
#>                       Estimate Std. Error   t value      Pr(>|t|)
#> (Intercept)         41.9143292 0.54314140 77.170198 1.816486e-278
#> sle_qua_c            0.6007436 0.18698805  3.212738  1.400240e-03
#> stu_hou_c           -0.7760596 0.09565223 -8.113346  3.902744e-15
#> sle_qua_c:stu_hou_c  0.3354037 0.03238114 10.357996  6.828773e-23
```

- **Coefficients for β_0 on M:**
  - β_{00} = 41.91, intercept
  - β_{01} = 0.60, linear effect
- **Coefficients for β_1 on M:**
  - β_{10} = -0.78, intercept
  - β_{11} = 0.34, linear effect

Moderation Effects



$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

$$\beta_0(M) = 41.91 + 0.60M$$

$$\beta_1(M) = -0.78 + 0.34M$$

Nonlinear Moderation Effects

- Moderated linear regression within the varying coefficients framework, extended with **nonlinear** effects:

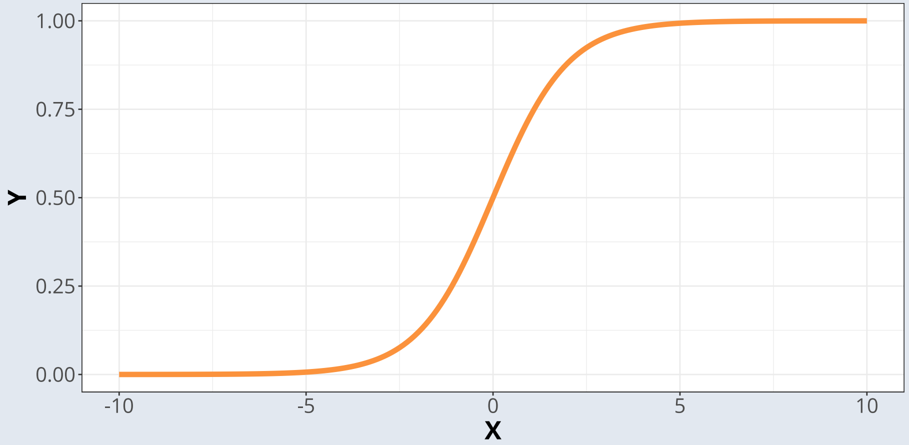
$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

- **Coefficients**
  - β_0 and β_1 can be moderated in a nonlinear regression regression by M
    - $\beta_0(M)$ = nonlinear regression of the intercept
    - $\beta_1(M)$ = nonlinear regression of the regression weight

Nonlinear Regression Models

Nonlinear in parameters

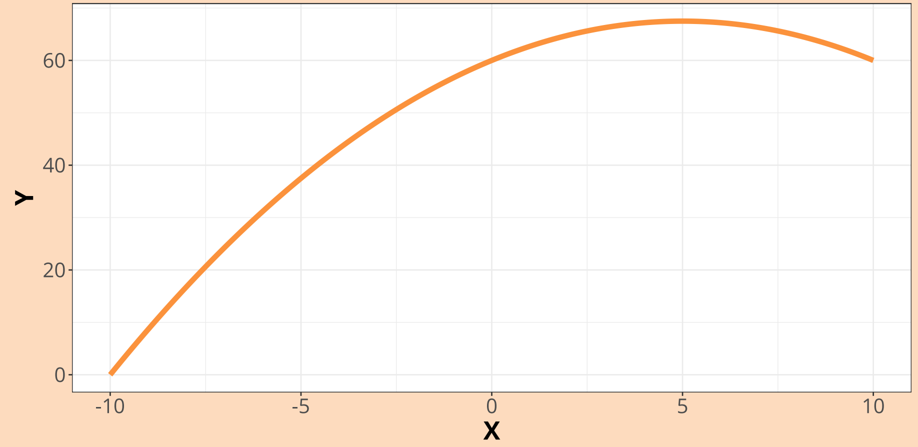
- Transforming X
- **Example:** logistic regression



$$P(Y=1|X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

Nonlinear in variables

- Transforming the range of X
- **Example:** polynomial regression

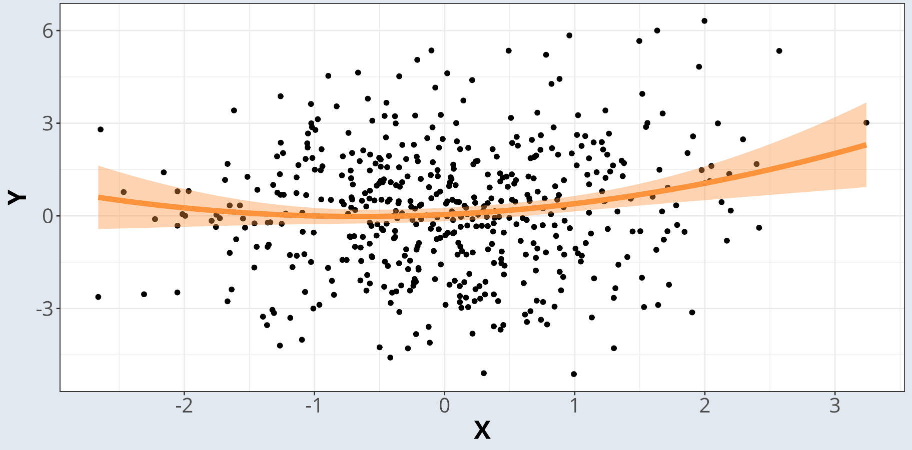


$$\hat{Y} = \beta_0 + \beta_1 X + \beta_2 X^2$$

Two Types of Nonlinear Models

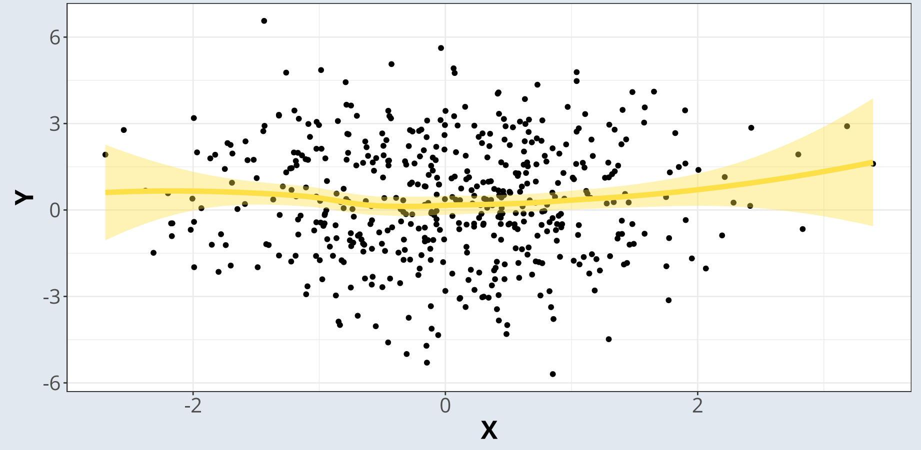
Global Models

- Whole range of X
- Parametric regression
  - Polynomial regression



Local Models

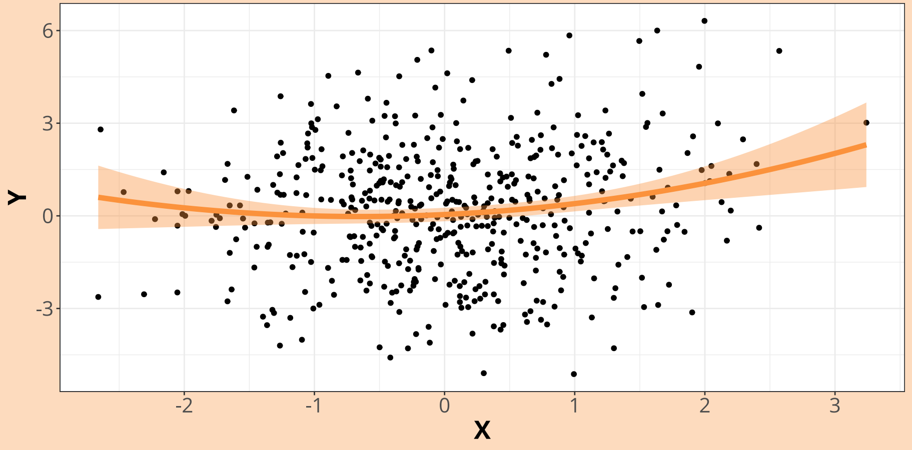
- Transforming the range of X
- Non-parametric regression
  - Local regression



Two Types of Nonlinear Models

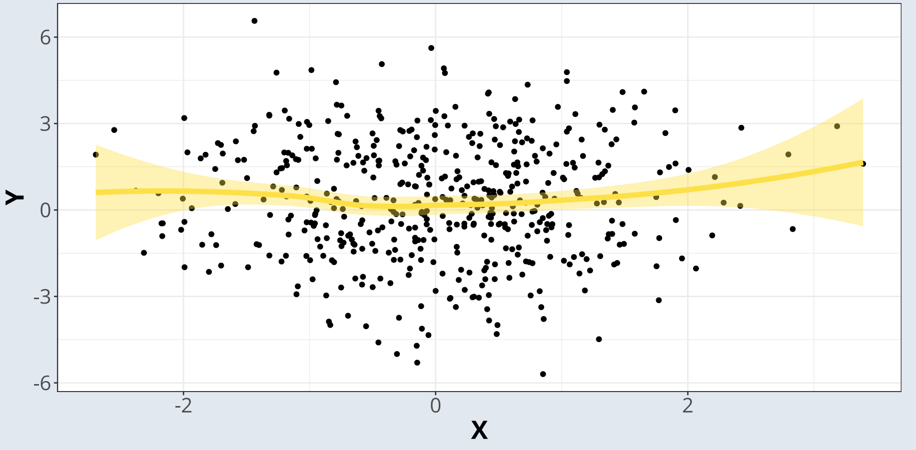
Global Models

- Whole range of X
- Parametric regression
  - Polynomial regression



Local Models

- Transforming the range of X
- Non-parametric regression
  - Local regression



Global Models

Assumption: single, unified (fixed) functional form across the **entire input space** to model relationship between the X and Y

- **Parametric models:** specific global relationship between X and Y
- **Advantages** (compared to local models):
  - **Simplicity and interpretability:** easier to understand and interpret (summarize data with few parameters)
  - **Computational efficiency:** less expensive compared to local models
  - **Extrapolation:** can be more reliable for extrapolating beyond the observed data
    - Depending on the correctness of the assumed global form!
- **Disadvantages** (compared to local models):
  - **Risk of misspecification** of potential nonlinear relationships
  - **Less flexibility** to detect local variations in the data

Quadratic Regression

- **Quadratic** regression with 1 predictor:

$$Y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

- **Coefficients:**
  - β_0 = zero-order coefficient, i.e., intercept
  - β_1 = first-order coefficient, i.e., linear effect
  - β_2 = second-order coefficient, i.e., quadratic effect

Quadratic Regression

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

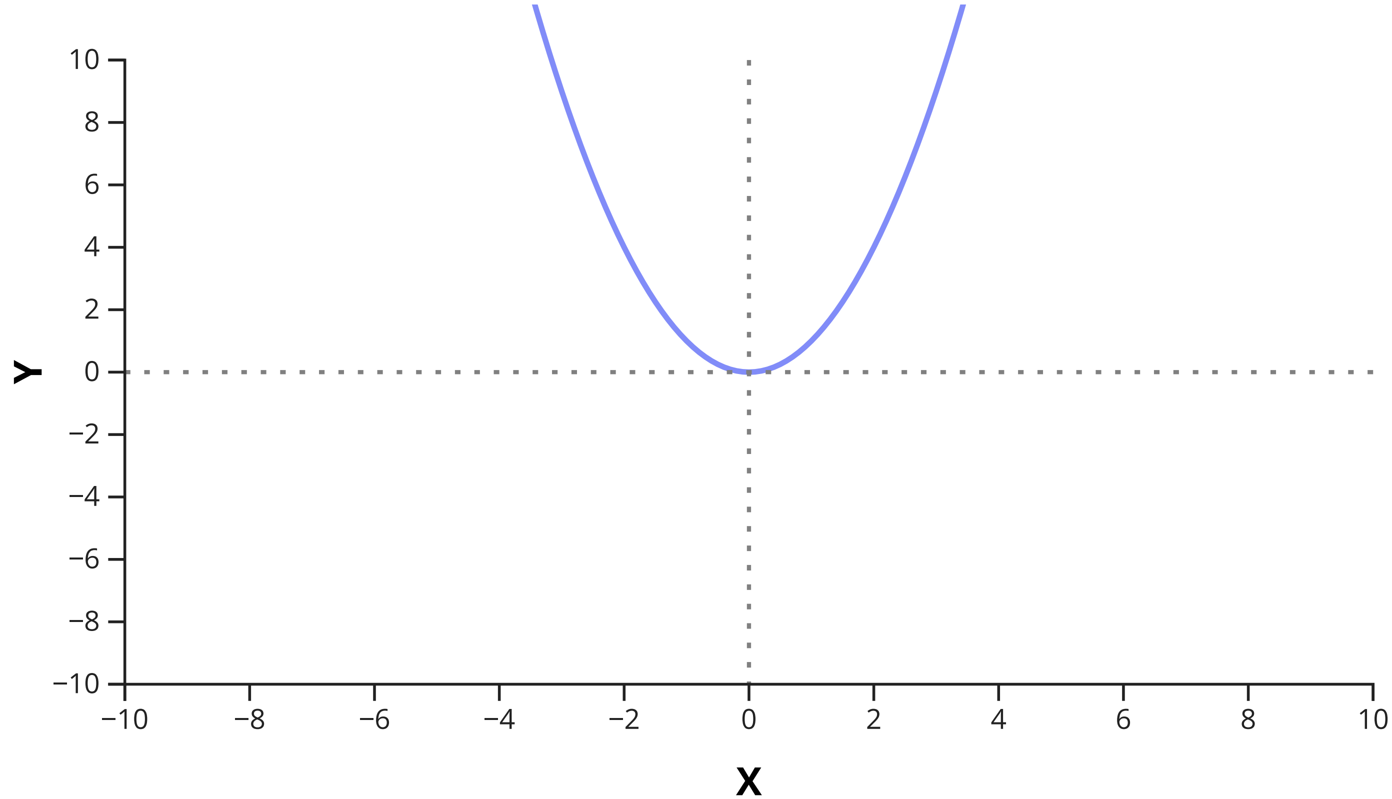
- β_2: **quadratic** coefficient
  - **Sign:** shape and direction
    - $\beta_2 > 0$: parabola opens upwards (U-shaped)
    - $\beta_2 < 0$: parabola opens downwards (inverted-U-shaped)
  - **Absolute size:** steepness
    - $|\beta_2| \to \infty$: narrower parabola
    - $|\beta_2| \to 0$: wider parabola

Quadratic Regression

β_2

1

Reset



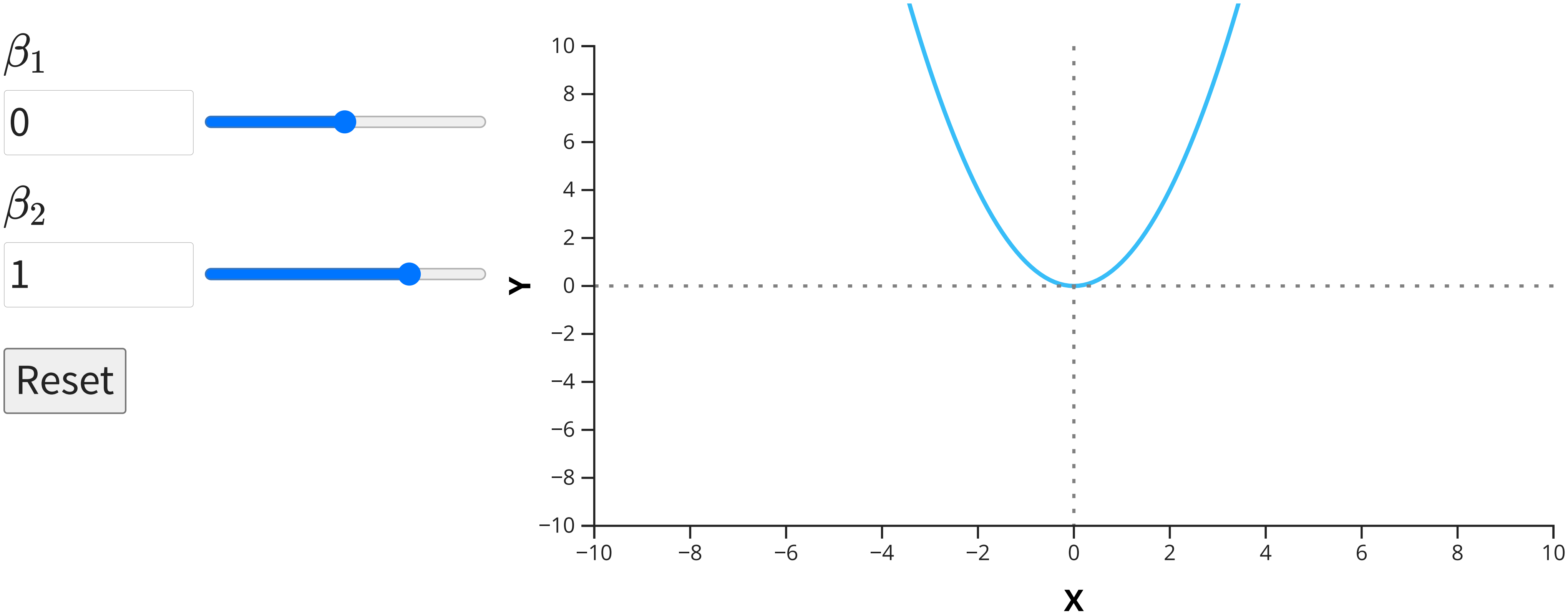
- **Parameter interpretation:**
  - $\beta_2 > 0$: parabola opens **upwards**

Quadratic Regression

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

- β_1: **linear** coefficient
  - **Tilt, horizontal vertex** position and **axis symmetry**
    - $\beta_1 > 0$: vertex shifts …
      - $\beta_2 > 0$: to the left of the Y-axis ($-X$-direction)
      - $\beta_2 < 0$: to the right of the Y-axis ($+X$-direction)
    - $\beta_1 < 0$: vertex shifts …
      - $\beta_2 > 0$: to the right of the Y-axis ($+X$-direction)
      - $\beta_2 < 0$: to the left of the Y-axis ($-X$-direction)
    - $\beta_1 = 0$: vertex lies on the Y-axis (X = 0)
  - **Slope** of the **tangent** at X = 0 (only meaningful for variables with natural 0 or centered variables)

Quadratic Regression



- **Parameter interpretation:**
  - $\beta_2 > 0$: parabola opens **upwards**
  - $\beta_1 = 0$: vertex lies at 0

Quadratic Regression

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \varepsilon$$

- β_0: **intercept**
  - **Vertical parabola position** and Y-axis intercept
    - $\beta_0 > 0$: parabola shifted upwards by β_0 units; positive Y-intercept
    - $\beta_0 < 0$: parabola shifted downwards by β_0 units; negative Y-intercept
    - $\beta_0 = 0$: parabola crosses origin; Y-intercept = 0

Quadratic Regression

β_0

0

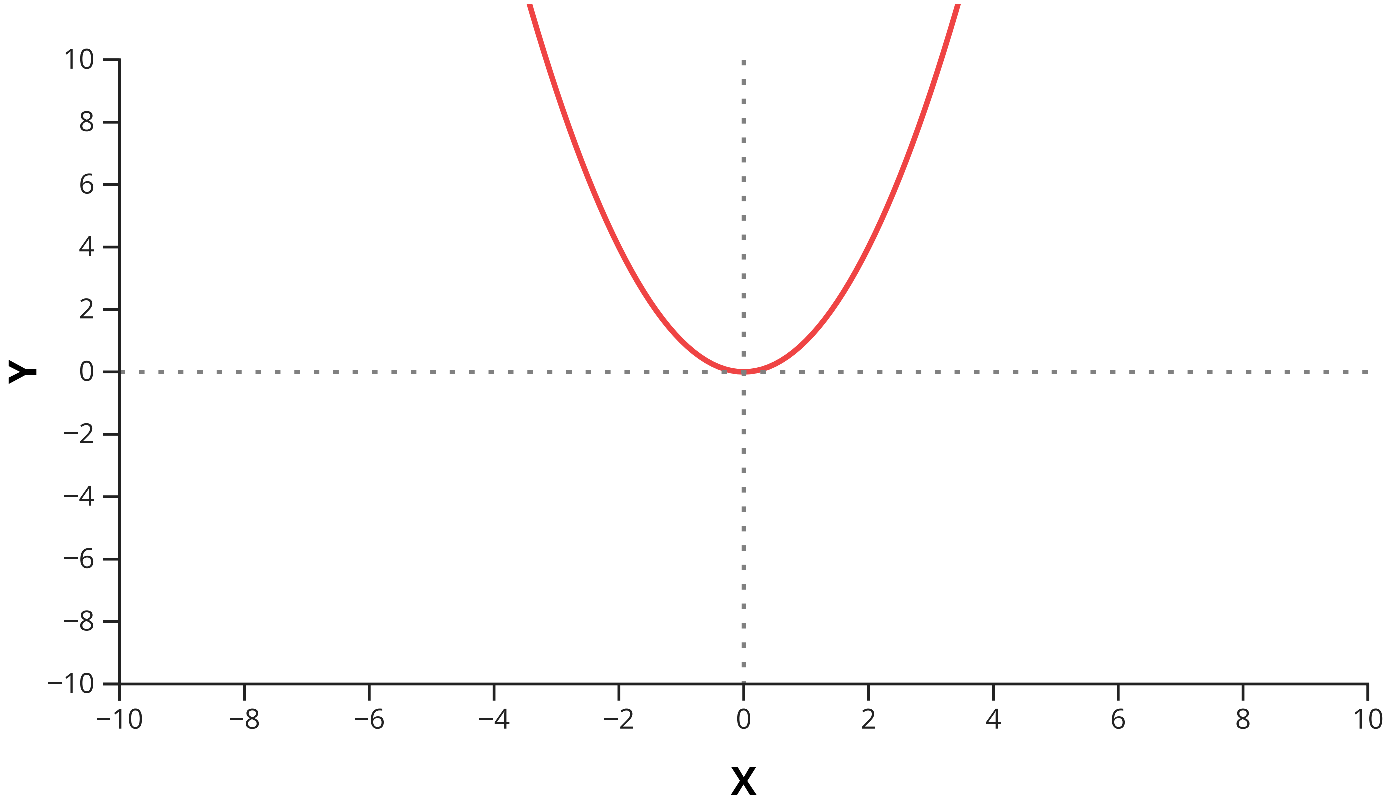
β_1

0

β_2

1

Reset



- **Parameter interpretation:**
  - $\beta_2 > 0$: parabola opens **upwards**
  - $\beta_1 = 0$: vertex lies at 0
  - $\beta_0 = 0$: intersects at 0

Example

Example 1.2: Academic Performance (`performance`)

You are studying the **nonlinear effect** of study hours X on academic performance Y.

```r
fit_qua <- lm(aca_per ~ 1 + stu_hou_c + I(stu_hou_c^2), data = performance)
summary(fit_qua)$coefficients
```

```
#>                    Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)     41.985517483 0.90236901 46.5281022 3.048493e-183
#> stu_hou_c       -0.742732706 0.10682403 -6.9528615  1.130989e-11
#> I(stu_hou_c^2) -0.007353772 0.02077193 -0.3540245  7.234706e-01
```

Obviously, the quadratic term is not supported. However, we could also compare the polynomial model to the model without a quadratic effect term:

```r
fit_lin <- lm(aca_per ~ 1 + stu_hou_c, data = performance)
anova(fit_lin, fit_qua)
```

```
#> Analysis of Variance Table
#>
#> Model 1: aca_per ~ 1 + stu_hou_c
#> Model 2: aca_per ~ 1 + stu_hou_c + I(stu_hou_c^2)
#>   Res.Df   RSS Df Sum of Sq      F Pr(>F)
#> 1    498 90550
#> 2    497 90528  1    22.829 0.1253 0.7235
```

Example

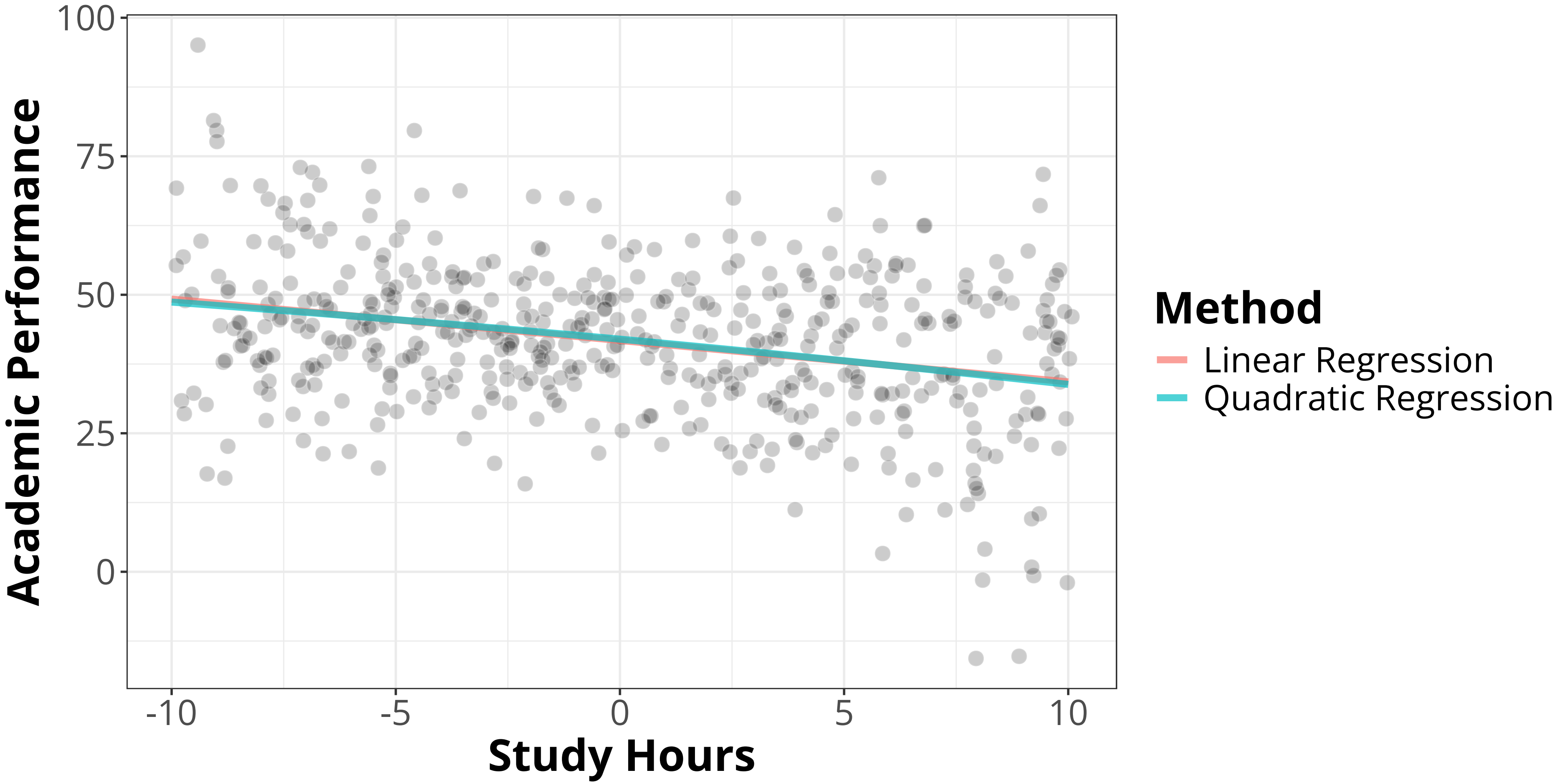
Model comparison: Models 1 (more restricted, e.g., linear, model) and Model 2 (less restricted, e.g., quadratic, model) can be compared via the F-test that checks the residual sums of squares RSS of two models:

$$F = \frac{(\mathrm{RSS}_1 - \mathrm{RSS}_2)}{(df_2 - df_1)} \Bigg/ \frac{\mathrm{RSS}_2}{(df_2)}$$

- **H_0:** Model 2 does not provide a significantly better fit than Model 1
- **Test statistic:** F-distributed with (df_1, df_2) degrees of freedom
  - **Significant:** Model 2 (more complex, e.g., quadratic) provides a significantly better fit than Model 1 (simpler, e.g., linear)
- **Prerequisite:** restricting the more complex model by fixing a parameter to 0 (e.g., the quadratic term β_0), i.e., gaining 1 df

Example

Comparing the two models:



Comparisons of the linear and the quadratic model (only the linear effect is supported)

Polynomial Regression

$$Y = \beta_0 + \beta_1 x + \beta_2 X^2 + \cdots + \beta_j X^{k-1} + \varepsilon$$

- **Degree** of the polynomial equation = value of the highest power (here $k - 1$)
  - **Example:** $Y = 5 + X + 3X^2 + 2X^3$ → 3rd degree polynomial
- **Highest-order term:** term with the highest power
- **Lower-order terms:** all other terms
- Caveats for **modeling** and **interpretation:**
  - **Higher order**: only interpretable if all lower terms are includes
    - Reflect specific level of curvature if all lower order terms are partialled out
  - **Lower order** (e.g., β_1 at X = 0): only interpretable if variable has a meaningful 0
    - No natural 0: centering required

Polynomial Regression

β_0

0

β_1

0

β_2

1

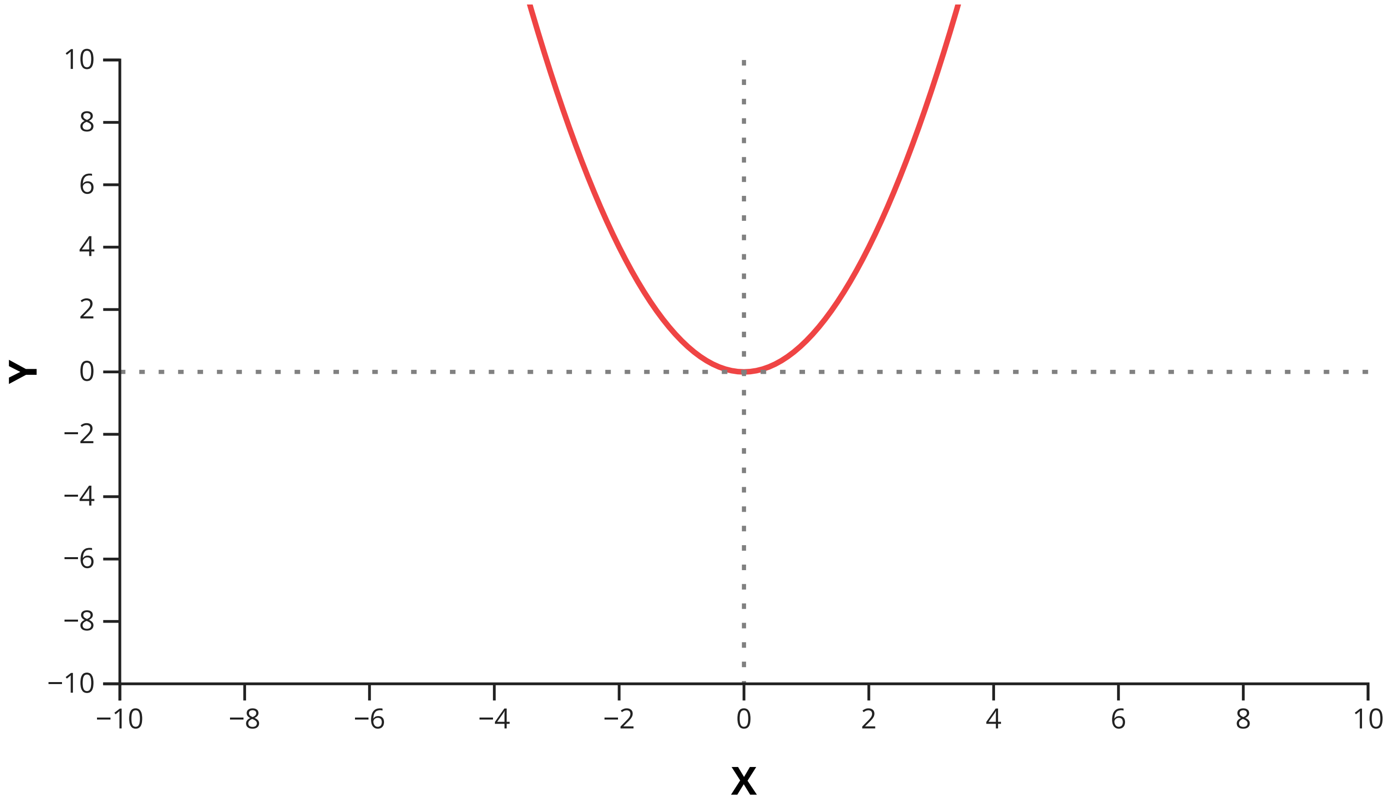
β_3

0

β_4

0

Reset



Moderated Nonlinear Regression

- Moderated linear regression with varying coefficients extended with **nonlinear**, i.e., quadratic moderator effects:

$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$
$$\beta_0(M) = \beta_{00} + \beta_{01}M + \beta_{02}M^2$$
$$\beta_1(M) = \beta_{10} + \beta_{11}M + \beta_{12}M^2$$
$$\hat{Y} = \beta_{10} + \beta_{11}M + \beta_{12}M^2 + \beta_{10}X + \beta_{11}MX + \beta_{12}M^2X$$

- **Coefficients**
  - β_0 and β_1 can be moderated in nonlinear, i.e., polynomial, fashion by M
  - β_{02} = quadratic effect of M on the intercept
  - β_{12} = quadratic effect of M on the regression weight

Example

Example 1.3: Academic Performance (`performance`)

You are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship in a nonlinear manner.

```r
fit <- lm(aca_per ~ 1 + sle_qua_c + I(sle_qua_c^2) +
          stu_hou_c + sle_qua_c : stu_hou_c + I(sle_qua_c^2) : stu_hou_c,
          data = performance)
summary(fit)$coefficients
```

```
#>                          Estimate Std. Error     t value      Pr(>|t|)
#> (Intercept)            42.3685991 0.68086185  62.2278945 6.653570e-236
#> sle_qua_c               0.5174748 0.15671305   3.3020530  1.029317e-03
#> I(sle_qua_c^2)         -0.0462456 0.06023384  -0.7677678  4.429920e-01
#> stu_hou_c               0.5468745 0.12040317   4.5420273  7.010025e-06
#> sle_qua_c:stu_hou_c     0.3021091 0.02715098  11.1270031  8.125187e-26
#> I(sle_qua_c^2):stu_hou_c -0.1512191 0.01030478 -14.6746519  1.000970e-40
```

- **Coefficient interpretation:** baseline and effects of M
  - `(Intercept)`: β_{00} = linear effect of M on β_0 (baseline)
  - `sle_qua_c`: β_{01} = linear effect of M on β_0
  - `I(sle_qua_c^2)`: β_{02} = quadratic effect of M on β_0 (not significant)

Example

Example 1.3: Academic Performance (`performance`)

You are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship in a nonlinear manner.

```r
fit <- lm(aca_per ~ 1 + sle_qua_c + I(sle_qua_c^2) +
          stu_hou_c + sle_qua_c : stu_hou_c + I(sle_qua_c^2) : stu_hou_c,
          data = performance)
summary(fit)$coefficients
```

```
#>                            Estimate Std. Error      t value      Pr(>|t|)
#> (Intercept)              42.3685991 0.68086185  62.2278945 6.653570e-236
#> sle_qua_c                 0.5174748 0.15671305   3.3020530  1.029317e-03
#> I(sle_qua_c^2)           -0.0462456 0.06023384  -0.7677678  4.429920e-01
#> stu_hou_c                 0.5468745 0.12040317   4.5420273  7.010025e-06
#> sle_qua_c:stu_hou_c       0.3021091 0.02715098  11.1270031  8.125187e-26
#> I(sle_qua_c^2):stu_hou_c -0.1512191 0.01030478 -14.6746519  1.000970e-40
```

- **Coefficient interpretation:** main effect and moderated effects of M
  - `stu_hou_c`: β_{10} = linear effect of M on β_1
  - `sle_qua_c:stu_hou_c`: β_{11} = linear effect of M on β_1
  - `I(sle_qua_c^2):stu_hou_c`: β_{12} = quadratic effect of M on β_1

Example

- Trimmed (via *p*-values), i.e., varying slope, model:

```r
fit <- lm(aca_per ~ stu_hou_c + sle_qua_c + sle_qua_c : stu_hou_c + I(sle_qua_c^2) : stu_hou_c,
          data = performance)
summary(fit)$coefficients
```

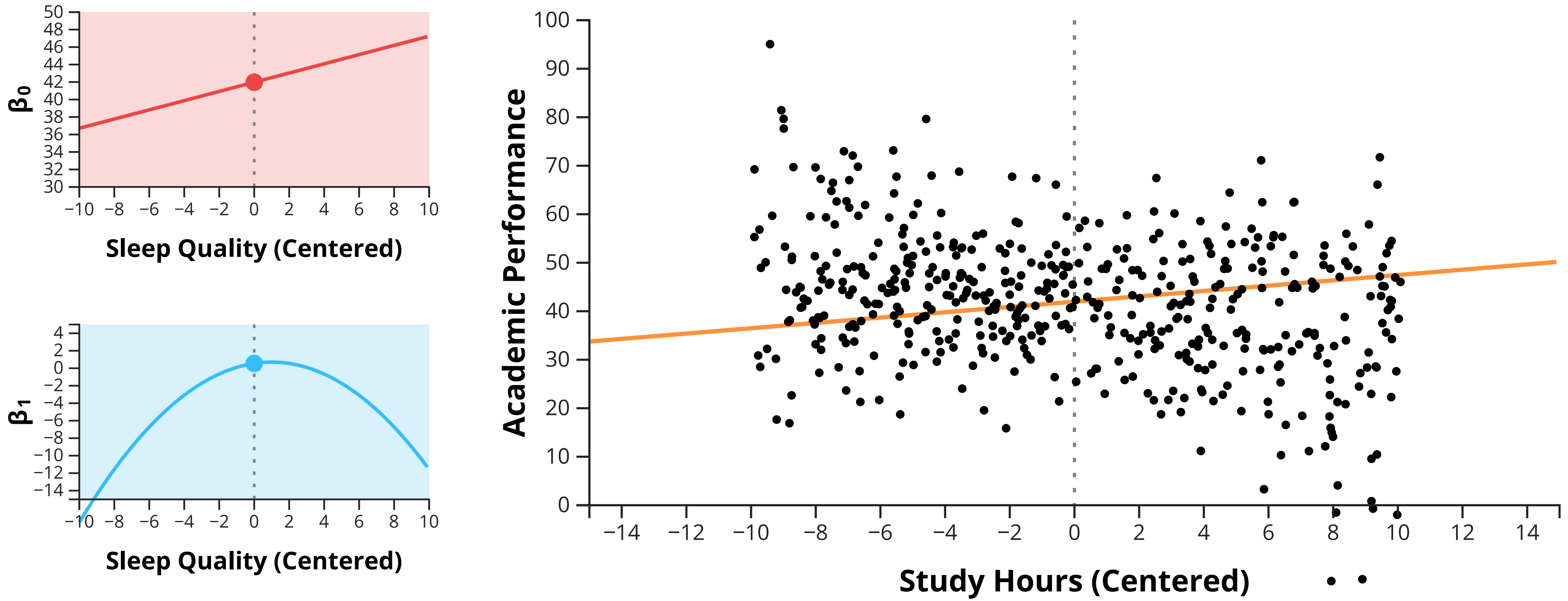
```
#>                              Estimate Std. Error    t value      Pr(>|t|)
#> (Intercept)                41.9788161 0.45349707  92.566896 1.119591e-314
#> stu_hou_c                   0.5485193 0.12033418   4.558300  6.505628e-06
#> sle_qua_c                   0.5265670 0.15620014   3.371105  8.072304e-04
#> stu_hou_c:sle_qua_c         0.3027654 0.02712627  11.161336  5.899566e-26
#> stu_hou_c:I(sle_qua_c^2) -0.1514876 0.01029457 -14.715285  6.401586e-41
```

- **Coefficient interpretation:** main effect and moderated effects of M
  - `(Intercept)`: β_{00} = academic performance for average M and X
  - `stu_hou_c`: β_{10} = linear effect of M on β_1
  - `sle_qua_c:stu_hou_c`: β_{11} = linear effect of M on β_1
  - `I(sle_qua_c^2):stu_hou_c`: β_{12} = quadratic effect of M on β_1

Example

- **Coefficient interpretation:**
  - `(Intercept)`: β_{00} = baseline academic performance for average M and X
  - `stu_hou_c`: β_{01} = base effect of sleep quality on academic performance
  - `stu_hou_c`: β_{10} = base effect of study hours on academic performance
  - `sle_qua_c:stu_hou_c`: β_{11} = linear moderation effect of sleep quality on the relationship between study hours and academic performance
  - `I(sle_qua_c^2):stu_hou_c`: β_{12} = quadratic moderation effect of sleep quality on the relationship between study hours and academic performance
    - indicating a diminishing return at higher sleep quality levels

Moderated Effects



$$Y = \beta_0(M) + \beta_1(M)X + \varepsilon, \varepsilon \sim \mathcal{N}(0, \theta_\varepsilon)$$

$$\beta_0(M) = 41.98 + 0.53M$$

$$\beta_1(M) = 0.55 + 0.30M + -0.15M^2$$

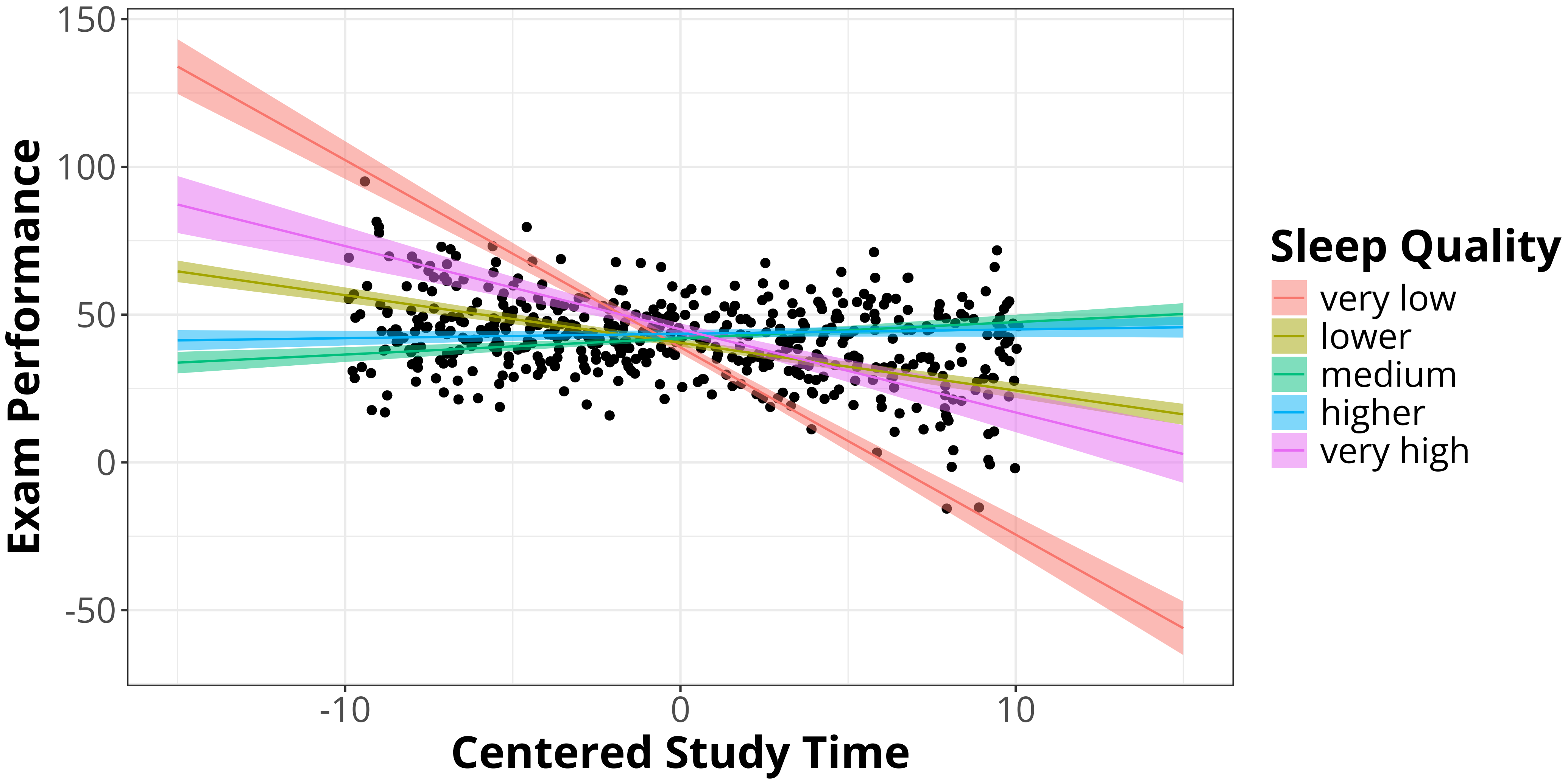
Example: Simple Slopes

```r
library(emmeans)
m_sq <- mean(performance$sle_qua_c)
sd_sq <- sd(performance$sle_qua_c)
focal_points <- m_sq + c(-2, -1, 0, 1, 2) * sd_sq
emtrends(fit, specs = ~ sle_qua_c | stu_hou_c, var = "stu_hou_c",
         at = list(sle_qua_c = focal_points))
```

```
#> stu_hou_c = 6.4e-16:
#>  sle_qua_c stu_hou_c.trend    SE  df lower.CL upper.CL
#>      -5.82          -6.336 0.304 495  -6.9327   -5.739
#>      -2.91          -1.613 0.115 495  -1.8380   -1.388
#>       0.00           0.549 0.120 495   0.3121    0.785
#>       2.91           0.148 0.110 495  -0.0679    0.364
#>       5.82          -2.814 0.322 495  -3.4467   -2.182
#>
#> Confidence level used: 0.95
```

- Very low `sle_qua`: increased `stu_hou` strongly decreases `aca_per`
- Low `sle_qua`: increased `stu_hou` decreases `aca_per`
- Average `sle_qua`: increased `stu_hou` enhances `aca_per`
- High `sle_qua`: increased `stu_hou` lightly enhances `aca_per`
- Very high: `sle_qua`: increased `stu_hou` decreases `aca_per`

Example: Ploting Interactions



Simple slopes for average sleep quality ± 1–2 *SD*

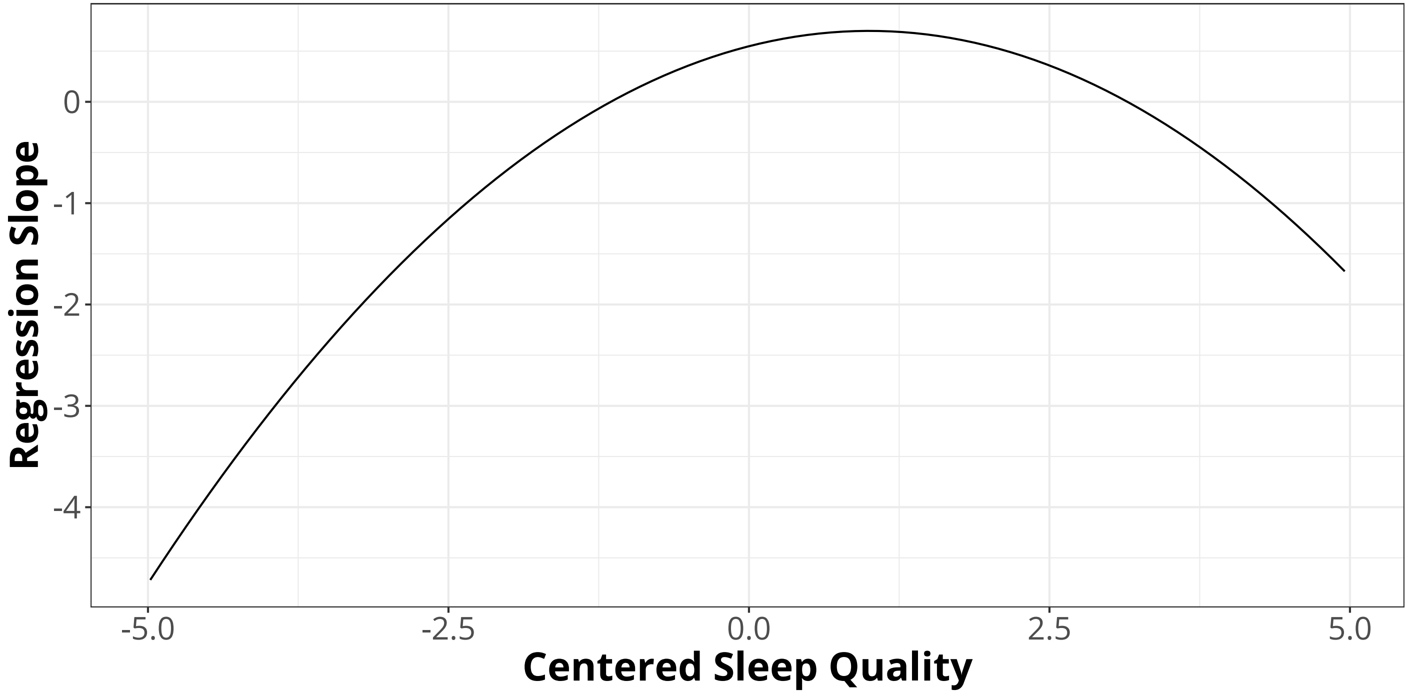
Visualization

- **Simple slope plots**: helpful for linear, but not for nonlinear moderation effects as the slopes may differ across the range of M → no sparse visualization
  - **Plot mapping:**
    - x-axis = X
    - y-axis = Y
    - grouping (e.g., color) = level of M (or potential interactions of moderators)
- **Parameter** (varying coefficient) **plots**: more flexible approach to capture the nonlinear moderation effects across the range of M → recommended for nonlinear moderations
  - **Plot mapping:**
    - x-axis = M
    - y-axis = coefficient, e.g., β_0 or β_1
    - grouping (e.g., color) = (level of) potential interactions of moderators

Example:Parameter Plots

- **Example:** regression coefficent $\beta_1(M)$ with manual prediction

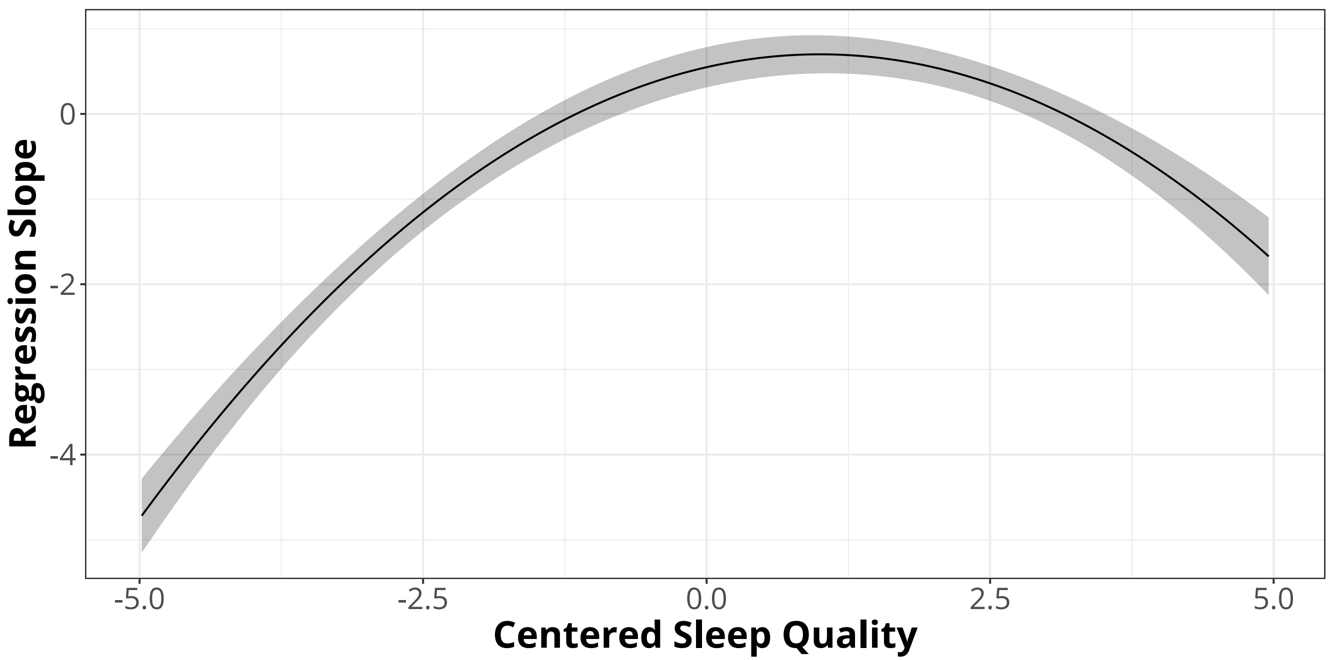
```r
b10 <- fit$coefficients[["stu_hou_c"]]
b11 <- fit$coefficients[["stu_hou_c:sle_qua_c"]]
b12 <- fit$coefficients[["stu_hou_c:I(sle_qua_c^2)"]]

ggplot(data = performance, mapping = aes(x = sle_qua_c)) +
  stat_function(fun = function(x) b10 + b11 * x + b12 * x^2) +
  labs(x = "Centered Sleep Quality", y = "Regression Slope")
```



Example:Parameter Plots

- **Example:** regression coefficent $\beta_1(M)$ with `emtrends()` (including CI)

```r
library(emmeans)
focal_points <- seq(min(performance$sle_qua_c), max(performance$sle_qua_c), length.out = 1000)
trend <- emtrends(fit, specs = ~ sle_qua_c | stu_hou_c, var = "stu_hou_c", at = list(sle_qua_c = focal_points

ggplot(data = as.data.frame(trend), mapping = aes(x = sle_qua_c, y = stu_hou_c.trend)) +
  geom_ribbon(aes(ymin = lower.CL, ymax = upper.CL), alpha = .3) +
  geom_line() +
  labs(x = "Centered Sleep Quality", y = "Regression Slope")
```



Model Complexity

- **Trade-off:**
  - **Too complex** models → potential overfit, i.e., fits the data to closely and captures noise rather than the underlying trend → poor generalization to new data
  - **Too simple** models → potential underfit, i.e., does not capture the underlying trend of the data
- **Model simplification:** approaches to avoid overfit
  - **Check significance:** non-significant (especially higher-order) terms could be dropped
  - **Model selection criteria:**
    - **Stepwise regression** (forward or backward) based on AIC or BIC
    - **Regularization techniques** (e.g., LASSO or ridge regression) to shrink coefficients of less important variables to 0
  - **Cross-validation:** ensure simpler model is robust and does not overfit

☕ Coffee Break

11:15–11:45

Session 2

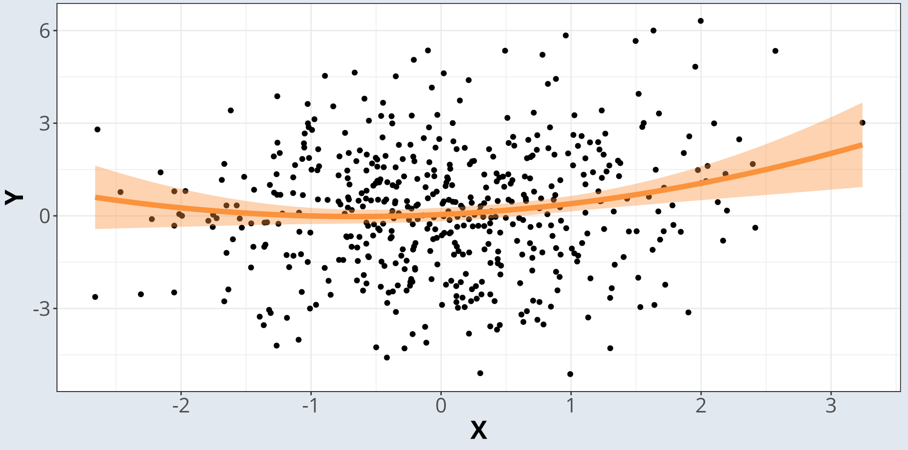
Local Regression

11:45–13:00

Two Types of Nonlinear Models

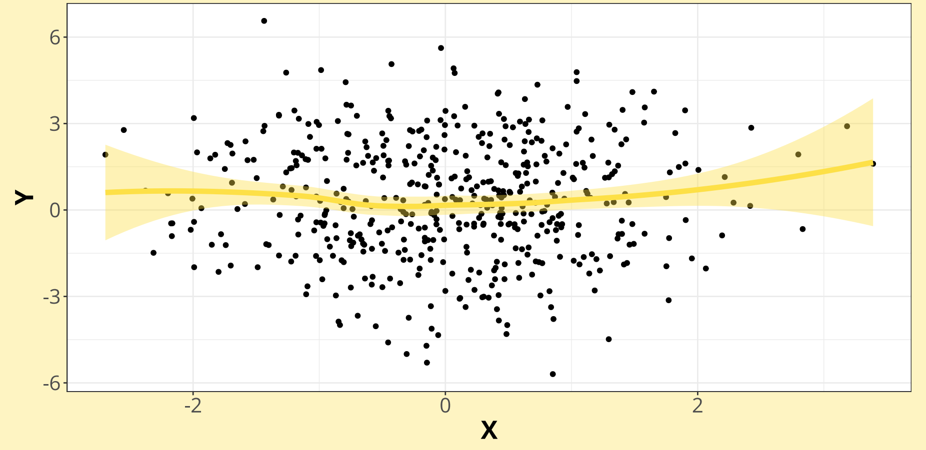
Global Models

- Whole range of X
- Parametric regression
  - Polynomial regression



Local Models

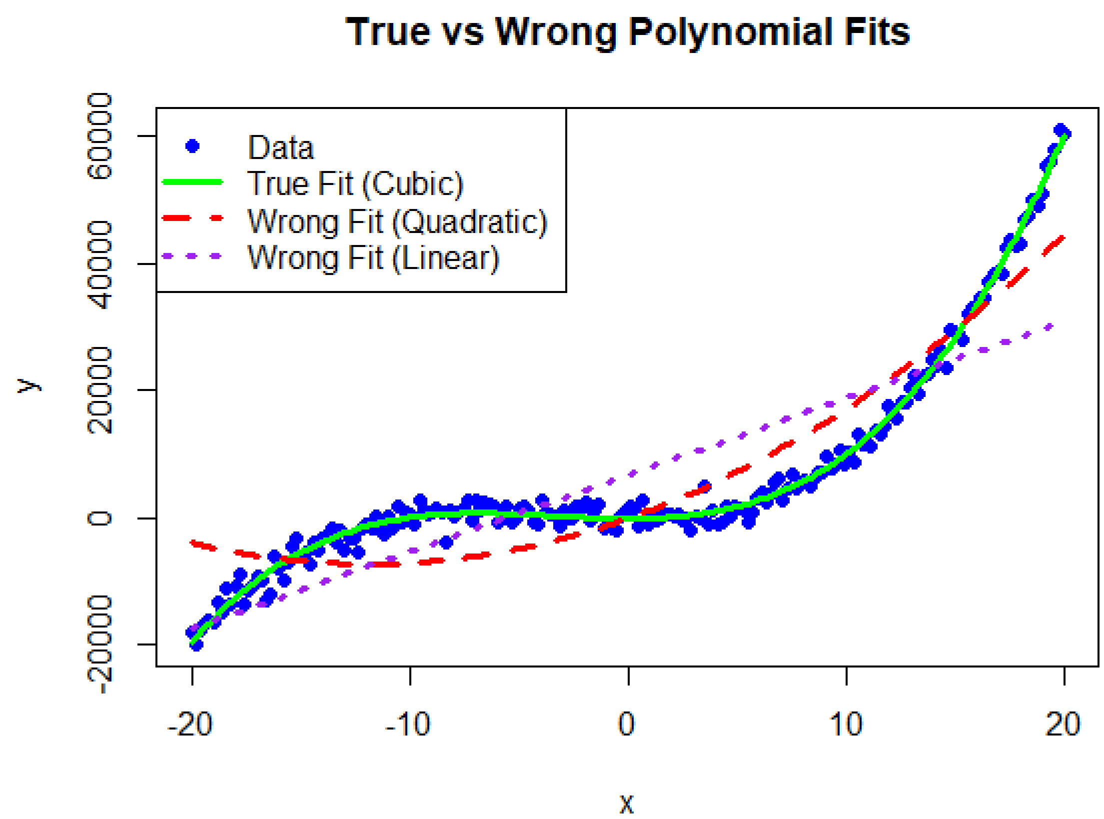
- Transforming the range of X
- Non-parametric regression
  - Local regression



Local Regression

Why non-parametric:

- Parametric non-linear function is a very effective way to fit the data (less parameter, higher statistical power)
- However, the correct functional form is usually unknown at the start of analysis
- Problem: high risk of fitting a curve that misrepresents the data's true structure



Wrong fit regression

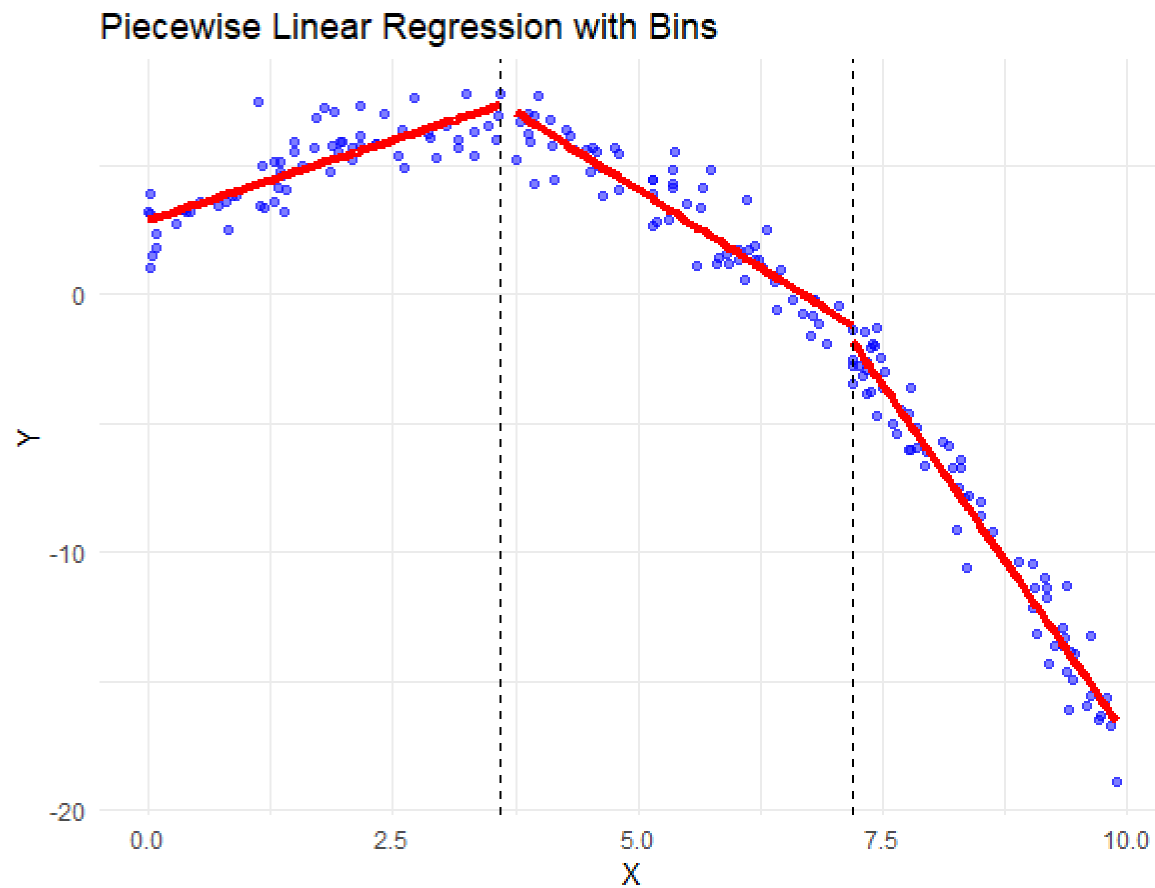
Local Model

In **global** models, the fitted relationship applies to the entire interval of independent variables and moderators.

- **Example:** quadratic regression

Local models (non-parametric) are based on the idea of dividing the observed range into a series of shorter spans and fitting different functions within each span (called bins).

- **Example:** piecewise linear regression

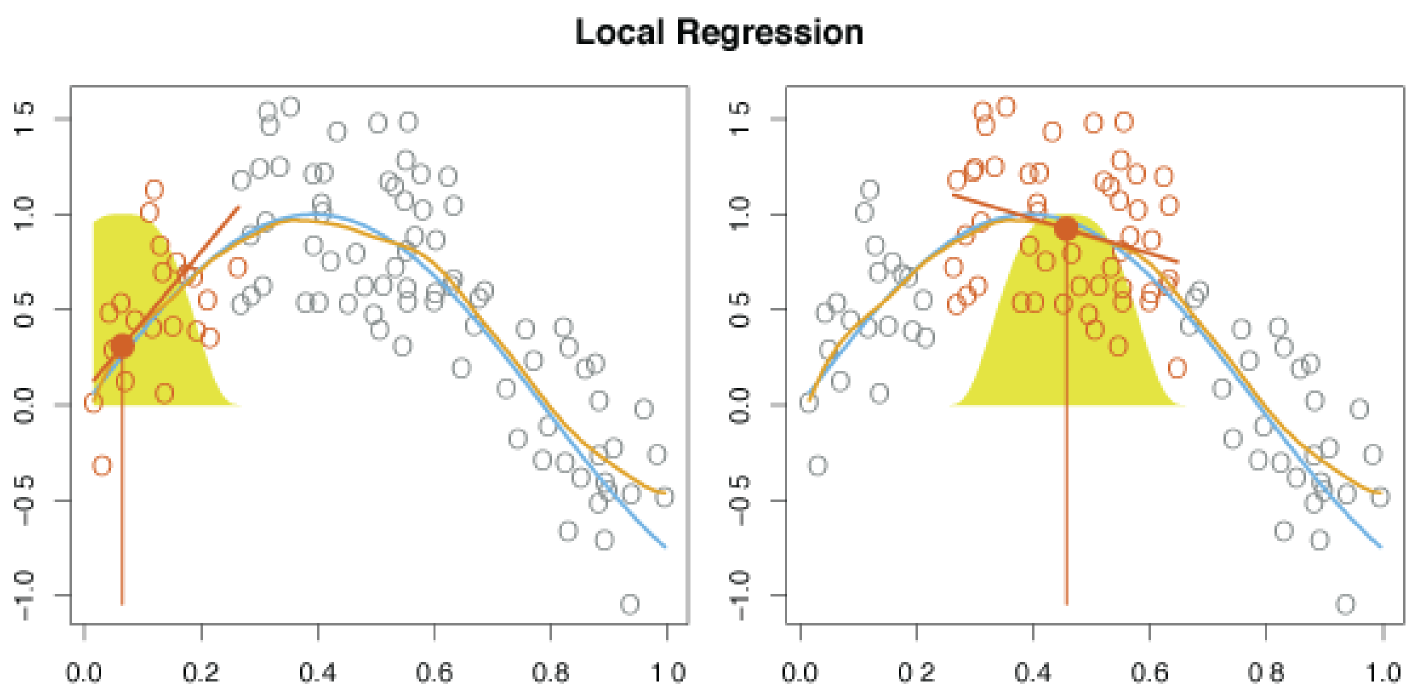


Piecewise Linear Regression

The Concept of “Data-Borrowing”

If there are many observations at each focal value of the moderator as bins, the regression can be modeled at each focal value.

- However, there is a single observation at each value
- but we can **borrow data** from nearby value



Local Regression (Rehabilitation Nursing Foundation Courses, 2024)

How to Borrow: Local Weighting

Naive idea: use the unweighted data from the n-th nearest neighbors and average them

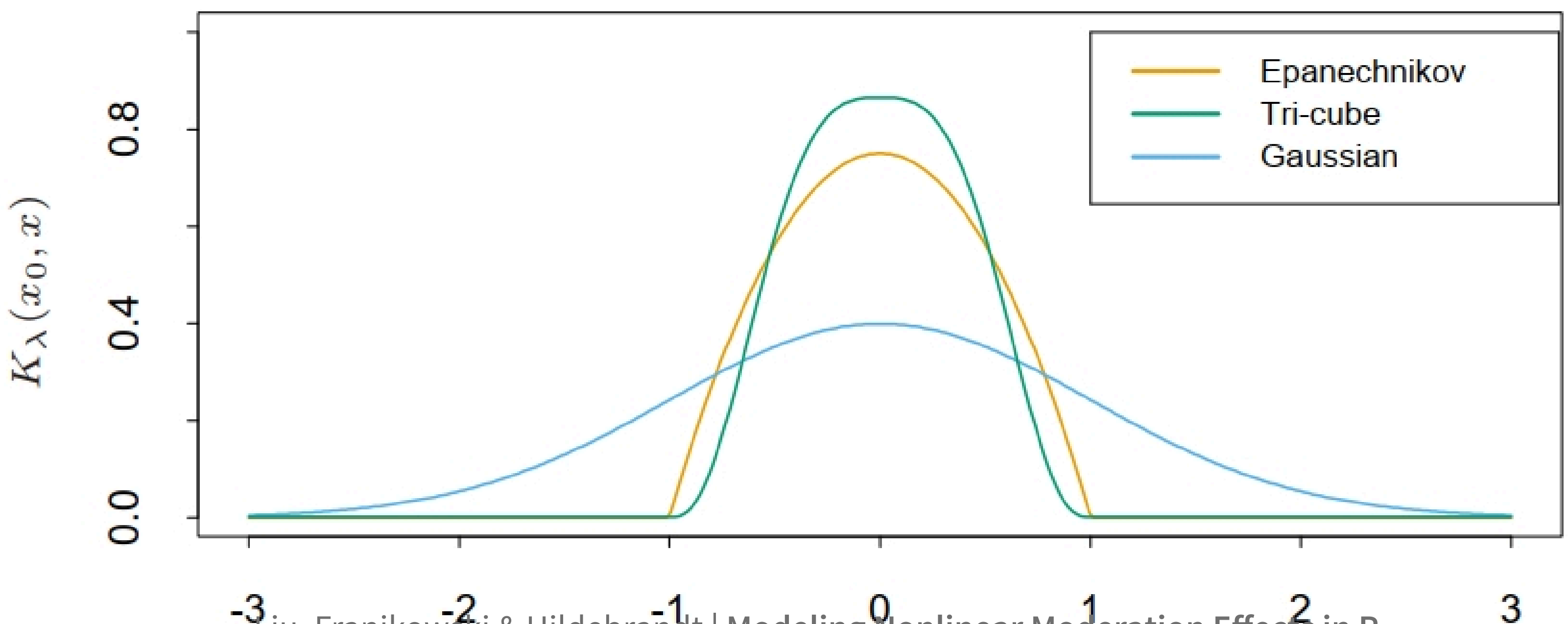
- Known as uniform kernel or k–nearest-neighbor kernel
- Same as the discretization of the continuous moderator variable into distinct groups

Better idea: the nearer, the higher the weight of the neighbors

- Known as kernel smoothing
- Typical choices of the weight function in regression and SEM are the Gaussian kernel and the Epanechnikov kernel (Robitzsch, 2023)
- F scatterplot smoothing, the tri-cube kernel was used (`loess()` in R)

How to Choose Local Weighting Function

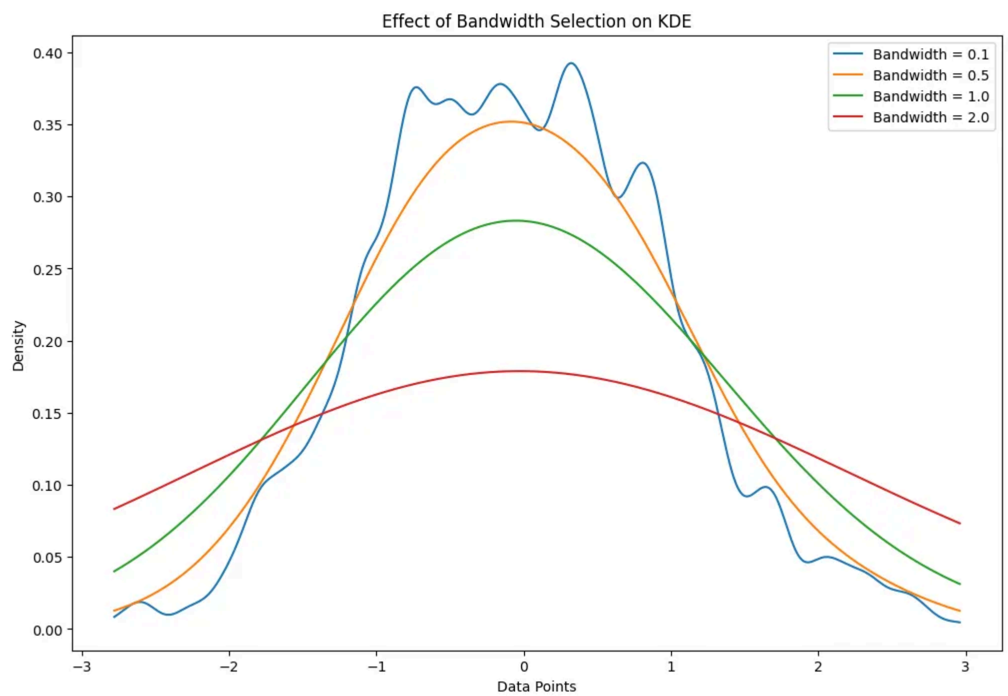
- There are absolute 0 weights in the Epanechnikov kernel and tri-cube kernel, but no absolute 0 weights in the Gaussian kernel
- Tri-cube kernel has smooth boundaries, but Epanechnikov kernel has sharp edges at the boundaries



The Concept of Bandwidth

The **bandwidth** defines the span or range of data points used to fit the model at a given focal value.

- A **smaller** bandwidth means fewer points are considered for each local model, leading to a more flexible and wiggly curve.
- A **larger** bandwidth includes more points, resulting in a smoother curve.



A comparison of different Bandwidth for local weighting (Raj, 2024)

How to Choose Bandwidth

Using the **Gaussian kernel** (we will focus in our workshop), an optimal bandwidth is given by the formula:

$$bw = hN^{-\frac{1}{5}}\sigma_A$$

where σ_A is the standard deviation of moderator.

However, different recommendations of h:

- h = 1.1 from most non-parametric regression literature (Silverman, 2018)
- h = 2 from multivariate context (Robitzsch, 2023)

For a single relationship (independent variable and dependent variable), we could use cross-validation or information criteria to choose the optimal h.

For multiple relationships, it would be difficult because the optimal h might be different for different relationships (multiscale problem, Li & Fotheringham, 2020).

Effective Sample Size

Effective sample size N refers to the number of data points significantly contributing to estimating the local model at a given focal value.

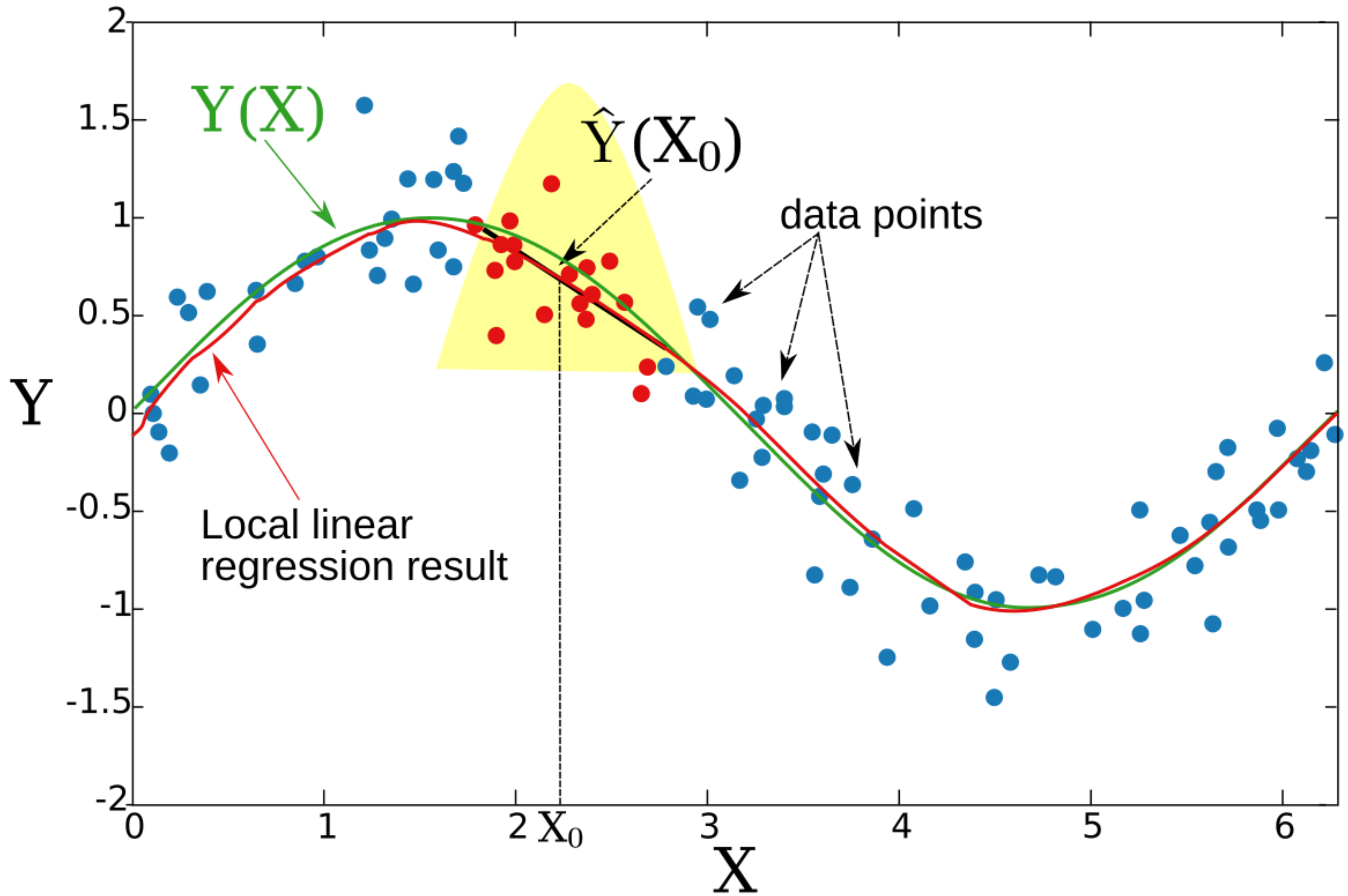
- **High** effective N: many data points contribute significantly to local model estimation at the given focal value. Occurs in regions with high data density using the same bandwidth.
- **Low** effective N: fewer data points contribute significantly. Occurs in regions with sparse data using the same bandwidth.

Effective N essentially acts as the local sample size and is related to the statistical power.

- At the boundary of the moderator, there is an inherent lack of data on one side; therefore, higher variability of estimation and larger confidence intervals

Local Linear Regression

The simplest local model is local linear regression, also known as **moving linear regression**.



Example of local linear regression (Wikipedia contributors, 2024)

Local Linear Regression: R Implementation

Usually used fit function, however I wouldn't say I like it:

```
loess(formula, data, span, degree,
    family = c("gaussian", "symmetric"), ...
)
?loess # to get a complete overview of argument
```

- **Arguments:**
  - `formula`: same as linear regression
  - `data`: same as linear regression
  - `span`: the parameter which controls the degree of smoothing (like h, but from 0 to 1)
  - `degree`: the degree of the polynomials to be used, normally 1 or 2
  - `family`: if "gaussian" fitting is by least-squares (analytic solution), and if "symmetric" a re-descending M-estimator used (iterations)

Local Linear Regression: R Implementation

Why I wouldn't say I like the `loess()` function: The `loess()` function …

- provides a convenient interface to fit local linear regression but carries out no inference (confidence interval, hypothesis tests, etc.)
- only supports continuous dependent variable
- can not include any other covariance

We could use `gam` package because the local linear regression is a special case of the Generalized Additive Model

- Please not use `mgcv` package because the local model was not supported.

Local Linear Regression: R Implementation

```r
gam::gam(
    y ~ lo(x, span, degree, ...), data,
    family, ...
)
```

- **Arguments:**
  - `lo()`: for loess smooth terms, s for smoothing splines, other same as generalized linear regression (Caution: only allowed parametric term interaction, but will not produce errors if you use non-parametric term interaction)
  - `data`: same as generalized linear regression
  - `span`: same like `loess()`, default `0.5`
  - `degree`: same like `loess()`, default `1`
  - `family`: link function

Local Linear Regression: R Example

Example 1.2: Academic Performance

You are studying the **nonlinear effect** of study hours X on academic performance Y.

```r
library(gam)
fit_loess01 <- gam::gam(aca_per ~ lo(stu_hou, span = 0.1, degree = 1), data = performance)

fit_loess01
summary(fit_loess01)$parametric.anova

pseudo_r_squared <- 1 - fit_loess01$deviance / fit_loess01$null.deviance
pseudo_r_squared
```

```
#> Call:
#> gam::gam(formula = aca_per ~ lo(stu_hou, span = 0.1, degree = 1),
#>     data = performance)
#>
#> Degrees of Freedom: 499 total; 480.8125 Residual
#> Residual Deviance: 87507.66
#> Anova for Parametric Effects
#>                                          Df Sum Sq Mean Sq F value    Pr(>F)
#> lo(stu_hou, span = 0.1, degree = 1)    1.00   9005  9004.7  49.477 6.939e-12 ***
#> Residuals                            480.81  87508   182.0
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> [1] 0.1210136
```

Local Linear Regression: R Implementation

```
#> Degrees of Freedom: 499 total; 480.8125 Residual
#> Residual Deviance: 87507.66

#> Anova for Parametric Effects
#>                                         Df Sum Sq Mean Sq F value    Pr(>F)
#> lo(stu_hou, span = 0.1, degree = 1)   1.00   9005  9004.7  49.477 6.939e-12 ***
#> Residuals                           480.81  87508   182.0
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

#> [1] 0.1210136
```

The nonlinear effect of study hours X on academic performance Y is significant, p < .001, the pseudo-R^2 is .12.

For `lm` objects, Efron's pseudo-R^2 will be equal to R^2. It should not be interpreted as R^2 for other models but can still be useful as a relative measure.

Local Linear Regression: Model Comparison

We may be interested in testing hypotheses such as whether there is any relationship between X and Y, or whether a local non-linear fitting improves the relationship more than a linear fitting.

- **Null model (M_0):**
  - $Y = \beta_0 + \varepsilon$
- **Linear model (M_1):**
  - $Y = \beta_0 + \beta_1 X + \varepsilon$
- **Local linear model (M_2):**
  - $Y = f(X) + \varepsilon$

These three models are nested sequentially.

Local Linear Regression: Model Comparison

The idea of the F-test (ANOVA) from conventional linear models also applies here (Andersen, 2009).

The formula for F is given by:

$$F = \frac{(\mathrm{RSS}_0 - \mathrm{RSS}_1)}{(\mathrm{EDF}_1 - \mathrm{EDF}_0)} \bigg/ \frac{\mathrm{RSS}_1}{(N - \mathrm{EDF}_1)}$$

where df is defined as:

$$df_1 = \mathrm{EDF}_1 - \mathrm{EDF}_0, df_2 = N - \mathrm{EDF}_1$$

- **Assumption:** normal distribution of residual and independently and identically distributed (i.i.d, Jacoby, 2000)
- **Monte-Carlo method** better (Bootstrap or Permutation)

Local Linear Regression: Example

Example 1.2: Academic Performance

You are studying the **nonlinear effect** of study hours X on academic performance Y.

```r
# Fit linear model
fit_null <- gam::gam(aca_per ~ 1,
    data = performance
)
# Model comparsion
anova(fit_null, fit_loess01, test = "F")
```

```
#> Analysis of Deviance Table
#>
#> Model 1: aca_per ~ 1
#> Model 2: aca_per ~ lo(stu_hou, span = 0.1, degree = 1)
#>   Resid. Df Resid. Dev     Df Deviance      F    Pr(>F)
#> 1    499.00      99555
#> 2    480.81      87508 18.188    12048 3.6396 7.401e-07 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can reject the null hypothesis that there is no relationship between X and Y → the relationship of study hours X on academic performance Y exists.

Local Linear Regression: Example

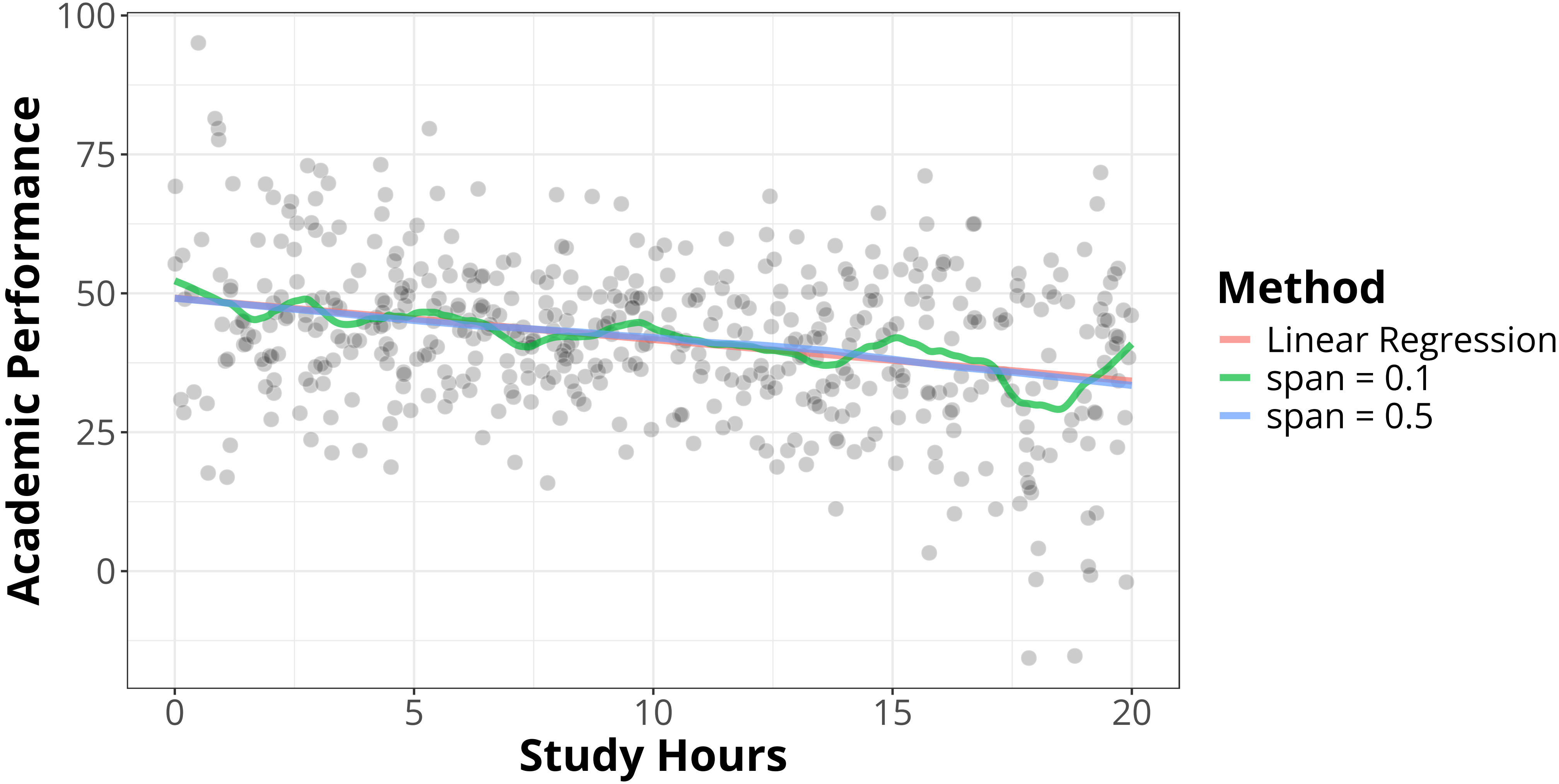
```r
# Fit linear model
fit_linear <- gam::gam(aca_per ~ stu_hou,
    data = performance
)
# Model comparsion
anova(fit_linear, fit_loess01, test = "F")
```

```
#> Analysis of Deviance Table
#>
#> Model 1: aca_per ~ stu_hou
#> Model 2: aca_per ~ lo(stu_hou, span = 0.1, degree = 1)
#>   Resid. Df Resid. Dev     Df Deviance      F Pr(>F)
#> 1    498.00      90550
#> 2    480.81      87508 17.188   3042.8 0.9727 0.4884
```

We can accept the null hypothesis of no improvement in fit in the local curve relative to the linear regression → linear effect of study hours X on academic performance Y

Local Linear Regression

Comparing different bandwidths and linear model:



Comparisons of different models and bandwidths

Non-Parametric Non-Linear Moderation

$$y = \beta_0(M) + \beta_1(M)X + \varepsilon$$

- **Explanation:**
  - M: moderator
  - $\beta_0(M), \beta_1(M)$: nonlinear functions of M for the intercept and for the slope

We could use the local model to estimate the $f(M)$. Different from the local linear model, we have local models in the coefficients (intercept, slope), varying with the moderator.

> **Example 1.3: Academic Performance**
>
> You are studying the **linear effect** of study hours X on academic performance Y, with sleep quality M moderating this relationship in a nonlinear manner.

Non-Linear Moderation: R Implementation

In R, there is no special package that could do non-linear moderation with the local model:

- `gam` package could do local linear regression easily but not support the interaction (moderation) with the non-parametric term
- `mgcv` package does not support the local model
- `loess()` function has so many disadvantages

However, when the smooth term is a function of geographic location (two dimensions), then the "varying coefficient models" are known as "geographic regression models" (GWR) in Geographical Research

- We could use geographic regression models and replace the geographic location with any moderator we are interested in.

Non-Linear Moderation: R Implementation

```r
GWmodel::gwr.basic(
    formula, data, regression.points,
    bw, kernel, F123.test, ...
)
GWmodel::ggwr.basic() # For generalized model, such as logistic link function
```

- **Arguments:**
  - `formula`: same as linear regression
  - `data`: must a `Spatial*DataFrame` object defined in package `sp`
  - `regression`.points: in which points of moderator the local model will run, default all points in x
  - bw: bandwidth used in the weighting function, same to h
  - `kernel`: “gaussian”, “tricube”, etc.
  - `F123.test`: If TRUE, conduct F-tests (Leung et al., 2000)

Non-Linear Moderation: R Example

Data Preparation for GWR:

```r
library(GWmodel)
library(sp)

# Only 1 moderator, so one dimension is set to 0
coordinates <- cbind(
    performance$sle_qua_c,
    rep(0, nrow(performance))
)
head(coordinates)

sp_perf <- SpatialPointsDataFrame(
    coords = coordinates,
    data = performance
)
```

```
#>           [,1] [,2]
#> [1,] -4.980803    0
#> [2,] -4.955877    0
#> [3,] -4.953756    0
#> [4,] -4.951650    0
#> [5,] -4.924384    0
#> [6,] -4.922735    0
```

Non-Linear Moderation: R Example

Fit GWR:

```r
gwr_model <- gwr.basic(aca_per ~ 1 + stu_hou_c,
    data = sp_perf, bw = 1.1, kernel = "gaussian",
    F123.test = T, parallel.method = "cluster"
)
gwr_model
```

```
#>    ***************Summary of GWR coefficient estimates:******************
#>                  Min.  1st Qu.   Median  3rd Qu.    Max.
#>    Intercept 39.54025 40.02320 42.59020 43.46667 44.0057
#>    stu_hou_c -3.56759 -1.22781 -0.32639  0.31433  0.6024
#>    ************************Diagnostic information*************************
#>    Number of data points: 500
#>    Effective number of parameters (2trace(S) - trace(S'S)): 10.37923
#>    Effective degrees of freedom (n-2trace(S) + trace(S'S)): 489.6208
#>    AICc (GWR book, Fotheringham, et al. 2002, p. 61, eq 2.33): 3762.535
#>    AIC (GWR book, Fotheringham, et al. 2002,GWR p. 96, eq. 4.22): 3751.928
#>    BIC (GWR book, Fotheringham, et al. 2002,GWR p. 61, eq. 2.34): 3294.802
#>    Residual sum of squares: 52268.21
#>    R-square value:  0.4749826
#>    Adjusted R-square value:  0.4638303
```

We could find the variance in both the intercept and slope of regression. However, is this variance significant or just an artifact of sampling?

Model Comparison with Local Moderation Effects

We could test whether a relationship between X and Y should be fixed globally (no moderation effect) or allowed to vary locally:

```
#>    ******************F test results of GWR calibration*******************
#>    ---F1 test (Leung et al. 2000)
#>     F1 statistic Numerator DF Denominator DF     Pr(>)
#>          0.58711          Inf            498 < 2.2e-16 ***
#>    ---F2 test (Leung et al. 2000)
#>     F2 statistic Numerator DF Denominator DF Pr(>)
#>          25.1265      -0.1459            498   NaN
#>    ---F3 test (Leung et al. 2000)
#>              F3 statistic Numerator DF Denominator DF     Pr(>)
#>    Intercept       6.4409     465.1255            Inf < 2.2e-16 ***
#>    stu_hou_c     120.2350     261.4704            Inf < 2.2e-16 ***
#>    ---F4 test (GWR book p92)
#>     F4 statistic Numerator DF Denominator DF     Pr(>)
#>          0.57723    489.62077            498 6.817e-10 ***
```

There are many options for F-tests, but usually, we focus on:

- $F1/F4$-test for the global model – same as global F-test in simple linear regression
- $F3$ test for each coefficient, same as t-test in simple linear regression

Local Coefficient Estimates Visualization

Now we could visualize the change trajectory of the coefficient (here just slope):

```r
# Extract coefficients and standard errors from the GWR model
coefficients <- gwr_model$SDF@data$stu_hou_c
se <- gwr_model$SDF@data$stu_hou_c_SE

# Create Confidence Intervals (CI)
upper_bound <- coefficients + 1.96 * se
lower_bound <- coefficients - 1.96 * se

# Create a dataframe for plotting
plot_data <- data.frame(
    sle_qua_c = performance$sle_qua_c,
    coefficients = coefficients,
    upper_bound = upper_bound,
    lower_bound = lower_bound
)
```

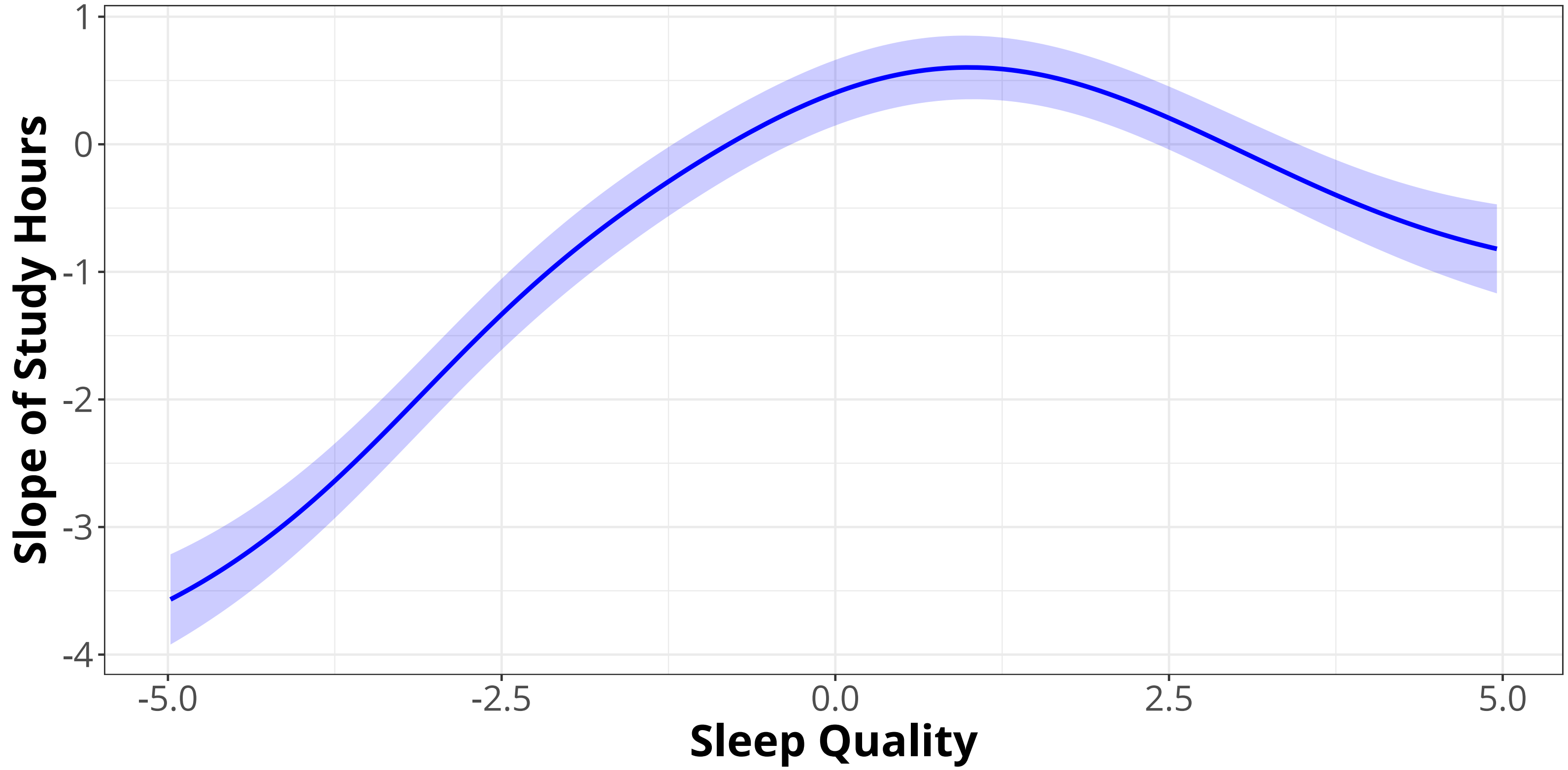
Local Coefficient Estimates Visualization

```r
library(ggplot2)
ggplot(
    data = plot_data,
    mapping = aes(x = sle_qua_c, y = coefficients)
) +
    geom_line(
        color = "blue",
        linewidth = 1
    ) +
    geom_ribbon(
        aes(ymin = lower_bound, ymax = upper_bound),
        alpha = 0.2,
        fill = "blue"
    ) +
    labs(
        x = "Sleep Quality",
        y = "Slope of Study Hours"
    )
```

Based on this plot, you could do a comparison between the values of the moderator.

- If the confidence intervals of two values do not overlap, it indicates a significant difference between them.

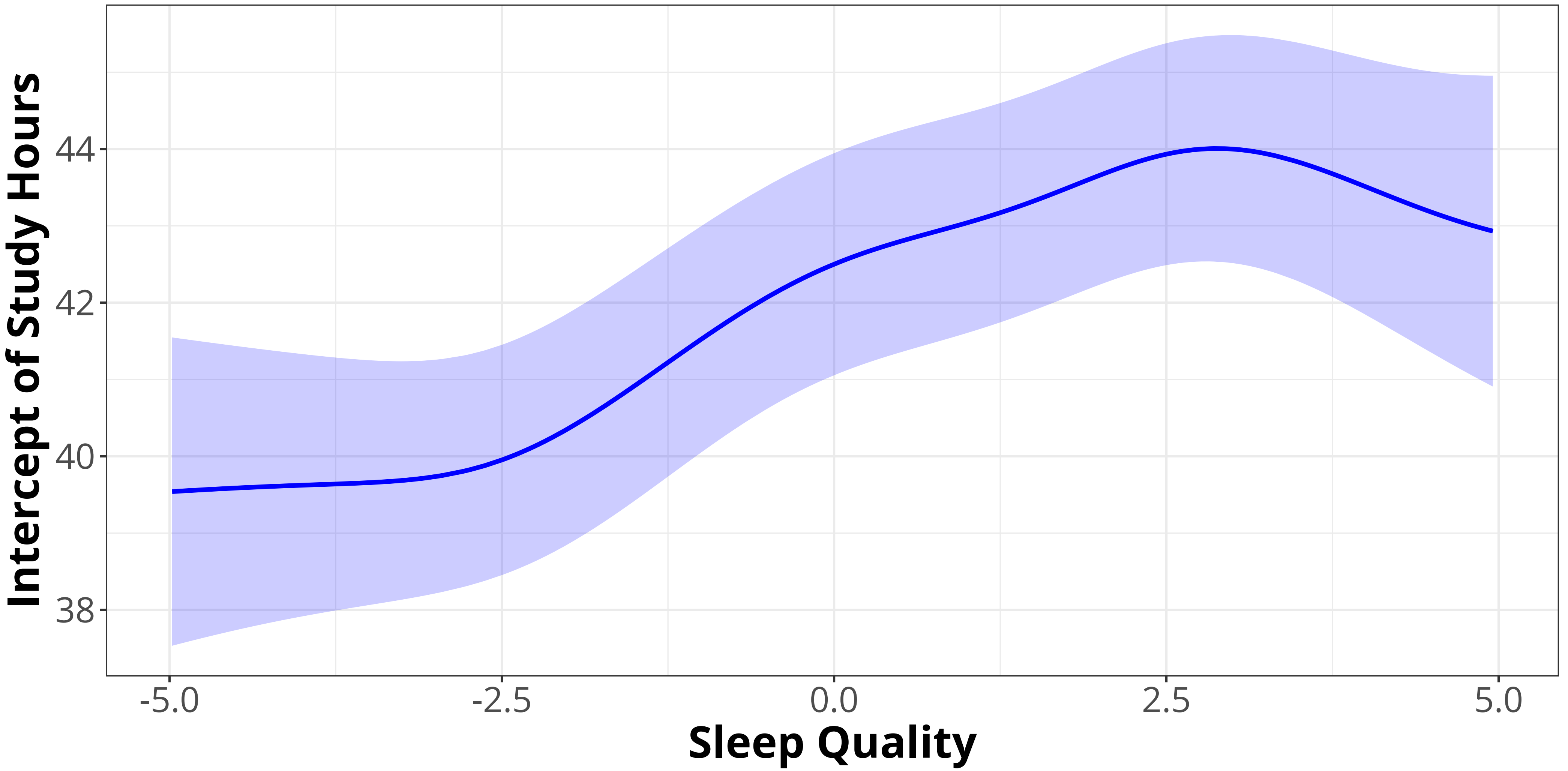
Local Coefficient Estimates Visualization



Confidence Interval for Regression Slope

The plot shows that the slope coefficient varies non-linearly with the moderator.

Varying Intercept Plot



Confidence Interval for Regression Intercept

Model Comparison with Local Moderation Effects

Again, we could test if a local non-linear model improves the fitting of moderation effects more than a linear model.

We used a **mixed (semiparametric)** GWR.

- **Semiparametric** = a model has both **parametric** and **non-parametric** components
  - GWR is a completely non-parametric model, we have moderation effects for both local intercept and slope.
- Therefore, each time, we can only compare one moderation effect

Next, we use slope coefficient as example

Model Comparison with Local Moderation Effects

- **Linear moderation model of slope (M_1):**
  - $Y = \beta_0(M) + \beta_1 X + \beta_3 MX + \varepsilon$
    - β_3: the linear moderation/Interaction effect for slope
    - $\beta_0(M)$: local (non-parametric model) nonlinear functions of M for intercept
- **Local non-linear moderation model (M_2):**
  - $Y = \beta_0(M) + \beta_1(M)X + \varepsilon$
    - $\beta_1(M)$: local (non-parametric model) nonlinear functions of M for slope

Effect of the Moderator M

The linear effect of the moderator M is absorbed into the local moderation effect of the intercept.

Model Comparison with Local Moderation Effects

```r
GWmodel::gwr.mixed(
    formula, data, regression.points,
    bw, kernel, fixed.vars, intercept.fixed, ...
)
```

- **Arguments:**
  - `formula`, `data`, `bw`: same as `gwr.basic()`
  - `regression.points`, kernel: same as `gwr.basic()`
  - `fixed.vars`: independent variables that are to be treated as global (no moderation)
  - `intercept.fixed`: logical, if `TRUE`, the intercept will be treated as global

Model Comparison with Local Moderation Effects

In our example:

```r
gwr_linSlo <- gwr.mixed(
    aca_per ~ stu_hou_c + stu_hou_c:sle_qua_c,
    data = sp_perf,
    fixed.vars = c("stu_hou_c", "stu_hou_c:sle_qua_c"),
    intercept.fixed = FALSE,
    bw = 1.1,
    kernel = "gaussian"
)
```

Model Comparison with Local Moderation Effects

However, this package can not calculate the F-test to compare one coefficient's local moderation and linear moderation effect.

- But, we could manually compute it (based on one option of F-test formula)

```
n <- 500
RSS0 <- gwr_linSlo$r.ss
RSS1 <- gwr_model$GW.diagnostic$RSS.gw
DF0 <- gwr_linSlo$df.used
DF1 <- gwr_model$GW.diagnostic$edf
F_value <- ((RSS0 - RSS1) / (DF1 - DF0)) / (RSS1 / (n - DF1))
F.p <- pf(F_value, DF1 - DF0, n - DF1, lower.tail = TRUE)
```

We must reject the null hypothesis ($p < .001$) of no improvement in fit in the local non-linear moderation relative to the linear moderation → the non-linear moderation effect.

Lunch Break

13:00–14:00

Recap 2

Structural Equation Models and Measurement Invariance

14:00–14:30

Structural Equation Models

- **Definition:**
  - Statistical technique that combines **factor analysis** and **multiple regression** (path analysis)
  - Allows for the modeling of complex relationships between observed (manifest) variables and latent (unobserved) constructs
- **Applications:**
  - Used in Psychology, Sociology, Economics, and other social sciences
  - Commonly applied to test-theoretical models to examine construct validity, mediation and moderation effects

Key Concepts

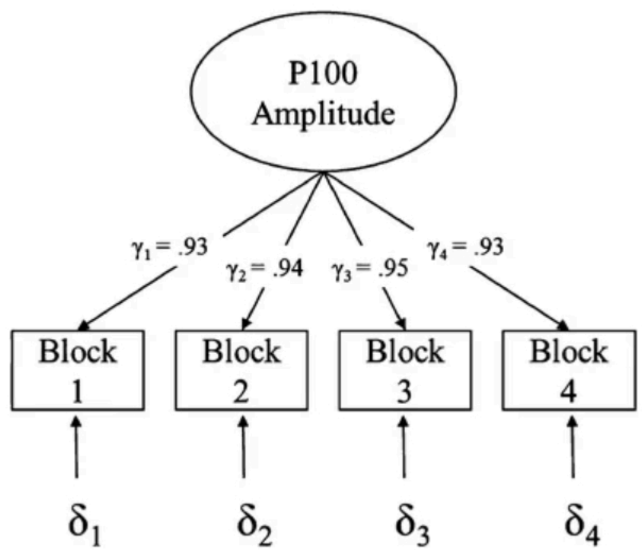
- **Variables:**
  - **Latent variables** (factors): unobservable constructs that are inferred from observed variables (indicators)
    - **Examples:** ability, attitude, action tendency, mood
  - **Observed variables** (items, indicators): directly measured variables used to estimate latent constructs
- **Model parts:**
  - **Measurement model:** specifies how latent variables are measured by observed variables (e.g., confirmatory factor analysis)
  - **Structural model:** defines relationships between latent variables, including direct and indirect effects

Components of SEM

- **Path diagrams:**
  - Visual representation of SEM models, showing the relationships between variables using arrows
    - **Single-headed** arrows: directional relationships (regression paths)
    - **Double-headed** arrows: correlations or covariances
- **Model estimation:**
  - Fitting the model to data to estimate path coefficients, variances, and covariances
  - Maximum Likelihood (ML) Estimation: most common method
- **Model fit:**
  - Assessed using the χ^2-test and fit indices such as:
    - **Goodness of fit:** CFI (Comparative Fit Index), TLI (Tucker-Lewis Index)
    - **Badness of fit:** RMSEA (Root Mean Square Error of Approximation), SRMR (Standardized Root Mean Residual)

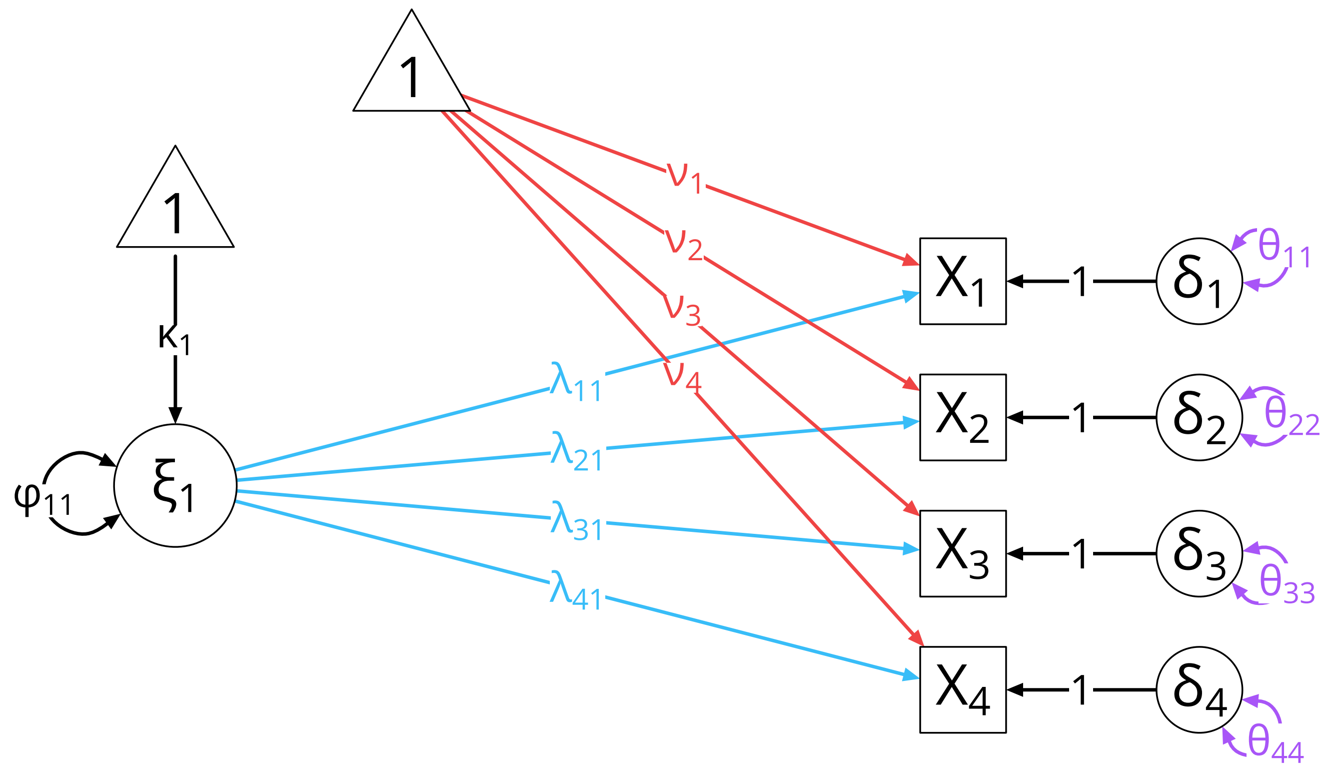
Measurement Models

- How can we **measure** something **unobservable**?
  - Build **measurement models** to represent the latent variable (“factor” / “construct”)
  - Latent traits can be measured using observed responses (on “items” / “indicators”)
- **Many options** of measurement models
  - Easiest option: Confirmatory Factor Analysis (CFA)



Measurement Model for ERP Components (Here, the P100 Amplitude, Kaltwasser et al., 2014)

Confirmatory Factor Analysis

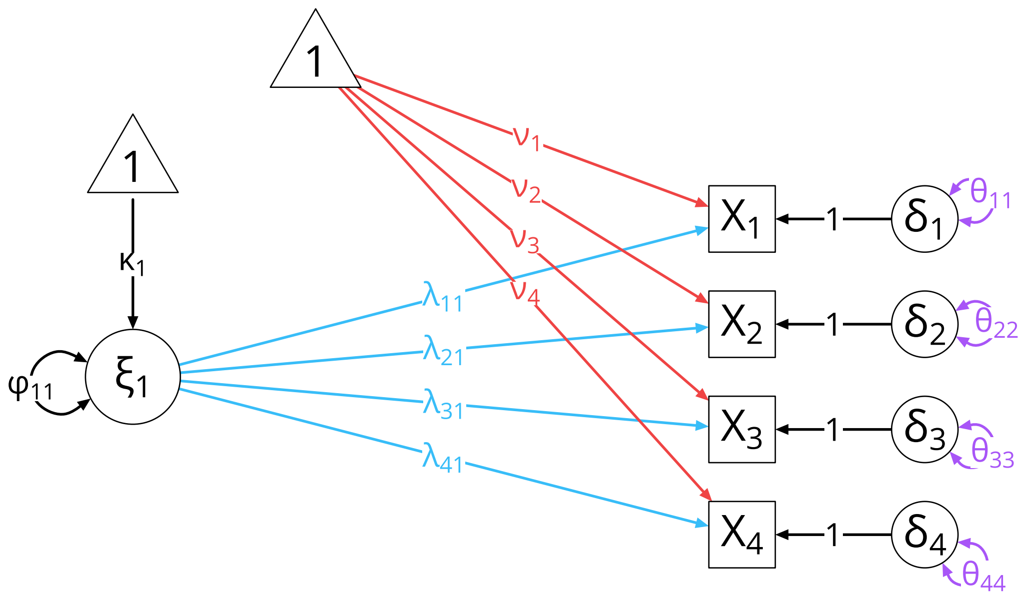


One-dimensional Measurement Model with 4 Indicators

- Regression equation to describe the measurement model:

$$X_i = \nu_i + \lambda_{ij}\xi_j + \delta_i, \delta_i \sim \mathcal{N}(0, \theta_{ii})$$

Confirmatory Factor Analysis

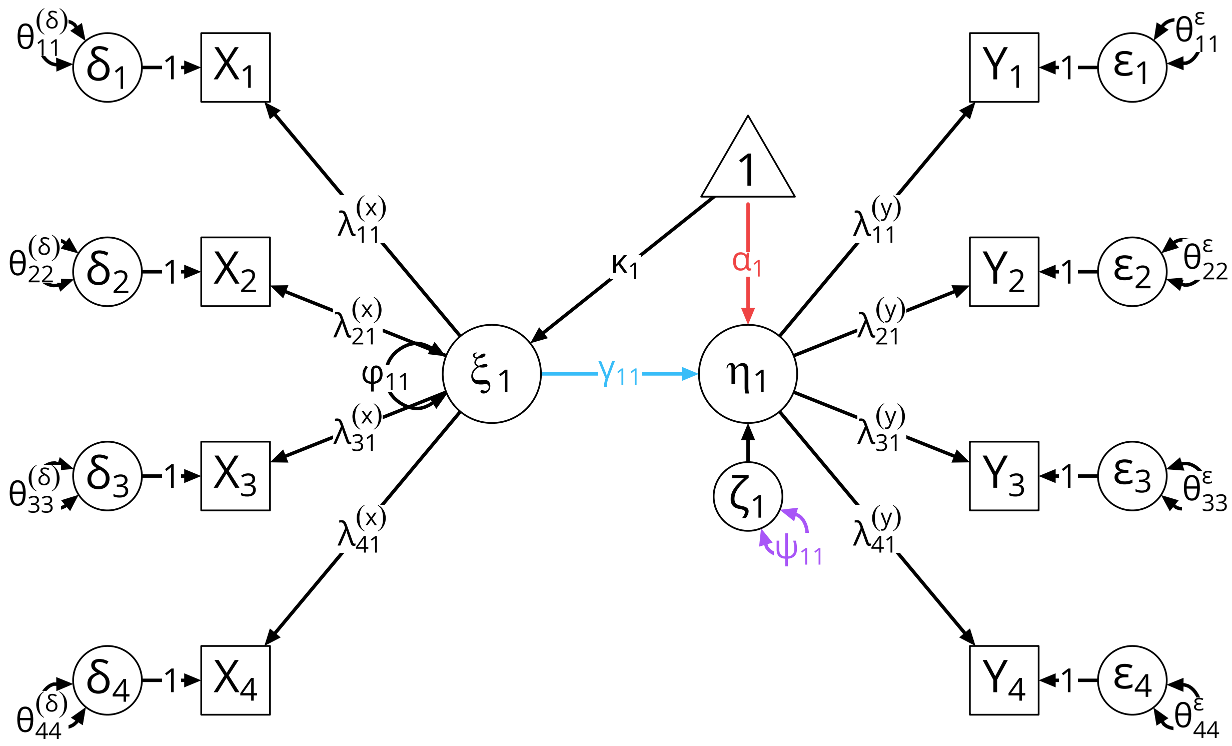


One-dimensional Measurement Model with 4 Indicators

$$X_i = \nu_i + \lambda_{ij}\xi_j + \delta_i, \delta_i \sim \mathcal{N}(0, \theta_{ii})$$

- **Variables:**
  - X_i = score on item i
  - ξ_j = score on factor j
  - δ_i = measurement error of item i
- **Coefficients:**
  - ν_i = intercept of item i
  - λ_{ij} = loading of item i on factor j
  - θ_{ii} = residual variance of item i
  - κ_j = latent mean of factor j
  - φ_{11} = variance of factor j

Structural Equation Modeling

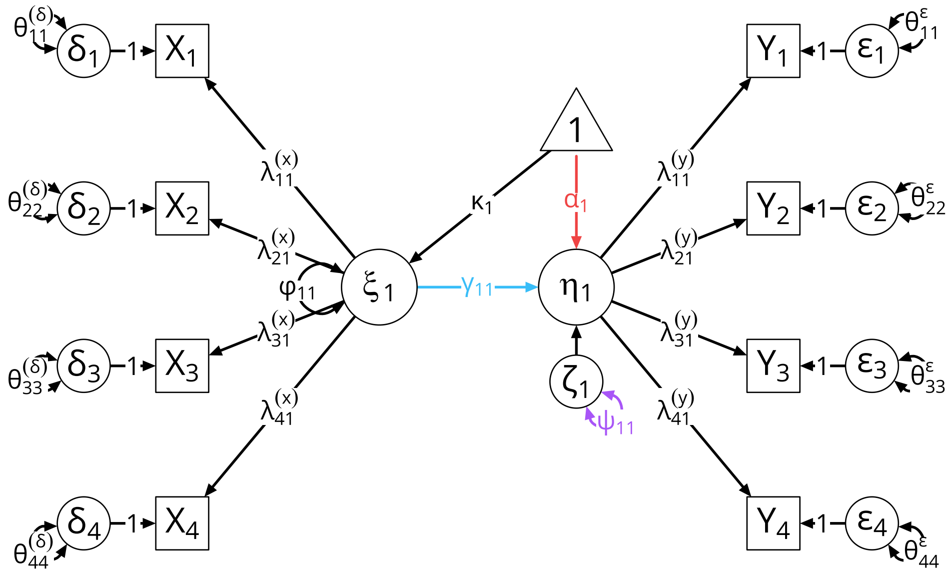


Structural Equation Model with 2 Latent Variables and 4 Indicators, Respectively (Intercepts of the Mesurement Models Were Omitted)

- (Latent) regression equations to describe the structural model:

$$\eta_i = \alpha_j + \gamma_{ij}\xi_i + \zeta_j, \zeta_j \sim \mathcal{N}(0, \psi_{ii})$$

Structural Equation Modeling

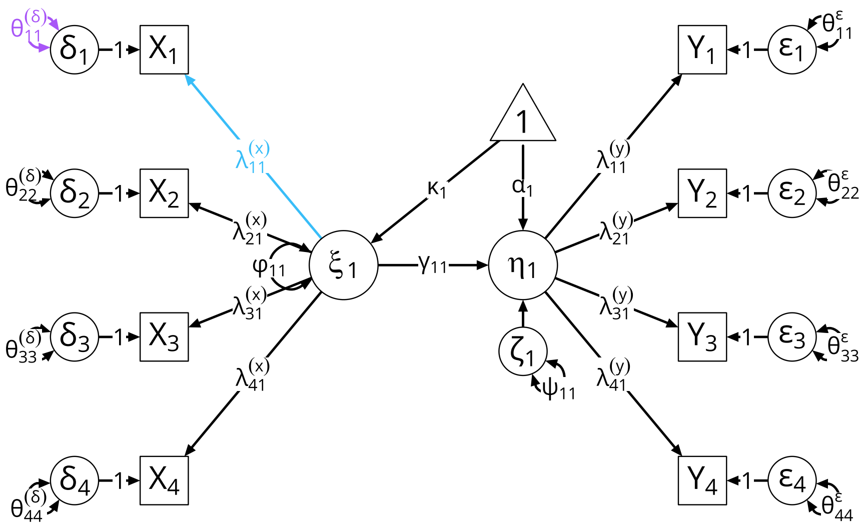


Structural Model

$$\eta_i = \alpha_j + \gamma_{ij}\xi_i + \zeta_j, \zeta_j \sim \mathcal{N}(0, \psi_{ii})$$

- **Variables:**
  - η_i = score on endogenous factor j
  - ξ_j = score on exogenous factor i
  - ζ_i = error of the latent regression
- **Coefficients:**
  - α_i = intercept of η_i
  - γ_{ij} = regression coefficient of η_i on ξ_j
  - ψ_{ii} = residual variance of η_j

Structural Equation Modeling

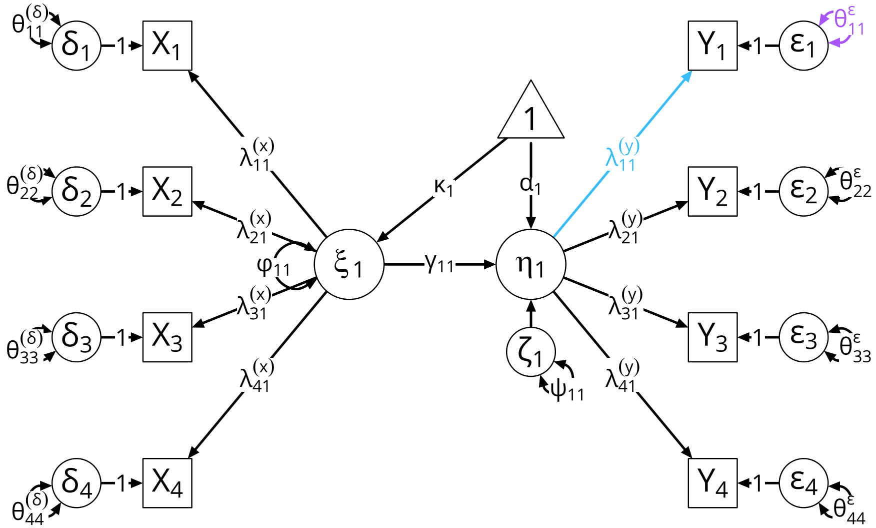


Measurement Model for X_k (Intercepts Not Depicted)

$$X_k = \nu_k^{(x)} + \lambda_{kj}^{(x)} \xi_j + \delta_i, \delta_i \sim \mathcal{N}(0, \theta_{kk}^{\delta})$$

- **Variables:**
  - X_k = score on item k
  - ξ_j = score on factor j
  - δ_k = measurement error of item k
- **Coefficients:**
  - $\nu_k^{(x)}$ = intercept of item k
  - $\lambda_{kj}^{(x)}$ = loading of item k on factor j
  - θ_{kk}^{δ} = residual variance of item k

Structural Equation Modeling



Measurement Model for Y_k (Intercepts Not Depicted)

$$Y_k = \nu_k^{(y)} + \lambda_{ki}^{(x)} \eta_i + \varepsilon_i, \varepsilon_i \sim \mathcal{N}(0, \theta_{kk}^{\varepsilon})$$

- **Variables:**
  - Y_k = score on item k
  - η_i = score on factor i
  - ε_k = measurement error of item k
- **Coefficients:**
  - $\nu_k^{(y)}$ = intercept of item k
  - $\lambda_{ki}^{(y)}$ = loading of item k on factor i
  - $\theta_{kk}^{\varepsilon}$ = residual variance of item kk

Dataset 2

Example 2.1: Parent Depression and Internalizing in Adolescents (`pure`)

We are studying the **correlation** between latent variables of parent depression (ξ_1) and adolescents' internalizing problems (ξ_2).

- **Variables:**
  - ξ_1 = `Depression`: measured by 9 items (`pdep{1-9}`)
  - ξ_2 = `Internalizing`: measured by 5 items (`cint{1-5}`)
- **Software**: `lavaan`

`lavaan`

- `lavaan` (`la`tent `va`riable `an`alysis): popular R package for SEM
- Allows for specification, estimation, and testing of SEMs
- Basic syntax: closely mirrors mathematical model and uses base R syntax

```r
model <- "
  # Measurement model
  xi1 =~ x1 + x2 + x3
  eta1 =~ y1 + y2

  # Structural model
  eta1 ~ xi1
"
```

Fitting an SEM Model with `lavaan`

- Steps to fit a model:
    1. Specify the measurement and structural models
    2. Fit the model using `lavaan::sem()` or `lavaan::cfa()`
    3. Inspect the output using `summary()` to get parameter estimates and fit indices

```r
model <- "
  # Measurement model
  xi1 =~ x1 + x2 + x3
  eta1 =~ y1 + y2

  # Structural model
  eta1 ~ xi1
"
fit <- sem(model, data = mydata)
summary(fit, fit.measures = TRUE)
```

Assessing Model Fit in `lavaan`

- **Key fit measures** and criteria (no "golden rule"):
  - χ^2-test: tests H_0 that the model fits the data perfectly (might be too sensible for large sample sizes and unreliable for small smaple sizes)
  - **CFI**: > .90 (Byrne, 1994), > .96 (Hu & Bentler, 1999)
  - **TLI**: > .90 (Byrne, 1994), > .95 (Schumacker & Lomax, 2004)
  - **RMSEA**: < .08 (Awang et al., 2015), <.06 (Hu & Bentler, 1999), < .05 (Byrne, 1994)
  - **SRMR**: < .08 (Byrne, 1994)

Fit Index Controversy

There is a lot of controversy about the use (and misuse) of these fit indices, so do not worry if the reviewer argues.

Assessing Model Fit in `lavaan`

CFI vs. TLI:

- **TLI** penalizes the complex model more than CFI (Gana & Broc, 2019)
- **TLI** and **CFI** do not vary much with sample size (Fan et al., 1999)
- **CFI** is less susceptible to the change of df (Shi et al., 2022)

> **Which One to choose?**
>
> There is a no "golden standard". However, the current practice is to report: $\chi^2 + df + p$-value, RMSEA, CFI, and SRMR (Kline, 2023). Try to make a holistic judgment based on a set of fit indices.

`lavaan` Syntax

Definitions:

Formula type	**Operator**	**Meaning**
latent variable definition	`=~`	is measured by
regression	`~`	is regressed on
intercept	`~ 1`	is regressed on 1
(residual) covariance	`~~`	is correlated with
constraints	`==`	is equal to

Example: Model Syntax

```r
library(lavaan)
load("data/pure.RData")

model <- "
  # Measurement models
  Depression =~ pdep1 + pdep2 + pdep3 + pdep4 + pdep5 + pdep6 + pdep7 + pdep8 + pdep9
  Internalizing =~ cint1 + cint2 + cint3 + cint4 + cint5

  # Latent covariance
  Depression ~~ Internalizing
"
fit <- sem(model = model, data = pure)
```

Latent correlation coefficient:

```r
summary(fit, standardized = TRUE, fit.measures = TRUE)
```

```
#> Covariances:
#>                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
#>   Depression ~~
#>     Internalizing     0.081    0.002   40.435    0.000    0.364    0.364
```

Example: Results

Fit indices (trimmed output):

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)                    0.965
#>   Tucker-Lewis Index (TLI)                       0.958
#>
#> Loglikelihood and Information Criteria:
#>
#>   Akaike (AIC)                              374829.186
#>   Bayesian (BIC)                            375057.748
#>   Sample-size adjusted Bayesian (SABIC)     374965.588
#>
#> Root Mean Square Error of Approximation:
#>
#>   RMSEA                                          0.062
#>   90 Percent confidence interval - lower         0.061
#>   90 Percent confidence interval - upper         0.064
#>   P-value H_0: RMSEA <= 0.050                    0.000
#>   P-value H_0: RMSEA >= 0.080                    0.000
#>
#> Standardized Root Mean Square Residual:
#>
#>   SRMR                                           0.024
```

Latent Variable Identification

A latent variable has no scale unit, because the latent variable was not measured, but estimated.

- Two common ways to define the scale of latent variables (factors, Kline, 2023):
    - **Unit loading identification:** loading of a **manifest variable** (scaling indicator) is fixed to **1**
    - **Unit variance identification:** variance of the **factors** is fixed to **1**

If we also model the **mean structure**:

- **Intercept** of a **manifest variable** (scaling indicator) is fixed to **0**
- **Mean** of the **factors** is fixed to **0**

Choice of scaling method has no effect on global model fit, the standardized solution, and unstandardized estimates for error terms (Kline, 2023)

- Unstandardized estimates for other model parameters depend on the method

Unit Variance Identification

In this workshop, we will use **UVI**, because in this way **both latent variables** will be z-standardized, i.e., **mean = 0** and **standard deviation = 1**

- Latent **covariance** coefficient → **correlation** coefficient
- Latent **unstandardized** → **standardized** regression coefficient

How to do that:

- **Option 1:** Use `lavaan` argument

```r
fit_uvi1 <- sem(
    model = model, data = pure,
    std.lv = TRUE
)
```

Unit Variance Identification

No UVI

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)
0.965
#>   Tucker-Lewis Index (TLI)
0.958
#>
#> Loglikelihood and Information Criteria:
#>
#>   Loglikelihood user model (H0)
-187385.593
#>   Loglikelihood unrestricted model (H1)
-184461.806
#>
#>   Akaike (AIC)
374829.186
#>   Bayesian (BIC)
375057.748
#>   Sample-size adjusted Bayesian (SABIC)
374965.588
#>
#> Root Mean Square Error of Approximation:
```

Option 1: Use `lavaan` argument

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)
0.965
#>   Tucker-Lewis Index (TLI)
0.958
#>
#> Loglikelihood and Information Criteria:
#>
#>   Loglikelihood user model (H0)
-187385.593
#>   Loglikelihood unrestricted model (H1)
-184461.806
#>
#>   Akaike (AIC)
374829.186
#>   Bayesian (BIC)
375057.748
#>   Sample-size adjusted Bayesian (SABIC)
374965.588
#>
#> Root Mean Square Error of Approximation:
```

Unit Variance Identification

In this workshop, we will use **UVI**, because in this way **both latent variables** will be z-standardized, i.e., **mean = 0** and **standard deviation = 1**

- Latent **covariance** coefficient → **correlation** coefficient
- Latent **unstandardized** → **standardized** regression coefficient

How to do that:

- **Option 2:** Use model syntax constraint

```
model_uvi <- "
  # Measurement models
  Depression =~ NA * pdep1 + pdep2 + pdep3 + pdep4 + pdep5 + pdep6 + pdep7 + pdep8 + pdep9
  Internalizing =~ NA * cint1 + cint2 + cint3 + cint4 + cint5

  # Latent correlation
  Depression ~~ Internalizing

  Depression ~~ 1 * Depression
  Internalizing ~~ 1 * Internalizing
"
fit_uvi2 <- sem(model = model_uvi, data = pure)
```

Unit Variance Identification

No UVI

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)
0.965
#>   Tucker-Lewis Index (TLI)
0.958
#>
#> Loglikelihood and Information Criteria:
#>
#>   Loglikelihood user model (H0)
-187385.593
#>   Loglikelihood unrestricted model (H1)
-184461.806
#>
#>   Akaike (AIC)
374829.186
#>   Bayesian (BIC)
375057.748
#>   Sample-size adjusted Bayesian (SABIC)
374965.588
#>
#> Root Mean Square Error of Approximation:
```

Option 2: Use model syntax constraint

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)
0.965
#>   Tucker-Lewis Index (TLI)
0.958
#>
#> Loglikelihood and Information Criteria:
#>
#>   Loglikelihood user model (H0)
-187385.593
#>   Loglikelihood unrestricted model (H1)
-184461.806
#>
#>   Akaike (AIC)
374829.186
#>   Bayesian (BIC)
375057.748
#>   Sample-size adjusted Bayesian (SABIC)
374965.588
#>
#> Root Mean Square Error of Approximation:
```

Measurement Invariance

Because we want to explore the moderated effect of structural coefficients, we first need to ensure that the construct is consistent, i.e., measurement invariant.

Measurement Invariance and Differential Item Functioning

The parameters (loadings, intercepts, and residual variances) of the measurement model are equal across the whole range of parameter values, M. In item response theory (IRT), it is called "differential item functioning" (DIF, Bauer, 2017).

Importance of measurement invariance (Maassen et al., 2023):

- Potential changes of measurement model can due to different sub-groups, time points, or experimental conditions
  - Therefore: always (even if only informally) examine measurement invariance when comparing across sub-groups, time points, or conditions; i.e., do not automatically assume measurement invariance

Measurement Invariance

Process to **test measurement invariance** in SEM (Putnick & Bornstein, 2016):

- **Simultaneous estimation** of SEMs for each sub-group, time point, or condition
- Parameters **freely** estimated or **constrained** to be equal across sub-groups, time points, or conditions
- **Model comparison** via different tests or indexes

Measurement invariance is essentially a matter of degree (Borsboom, 2006):

- Large samples increase the chance of finding statistically significant differences
  - Consider if differences in the measurement model are substantively meaningful, not just whether they are statistically significant

Testing Measurement Invariance

Measurement invariance can be regarded as a specific type of moderation that impacts only measurement model parameters (i.e., factor loading, intercepts, and residual variances, Hildebrandt et al., 2016)

- **Types of moderator variables** M:
  - **Categorical**: multi-group SEM (usual approach)
  - **Continuous**: moderated SEM (varying coefficient approach)
- **Procedure:** successive **restricting** free parameters to be **equal** across the range of M
  - **Model comparison** via different fit indices (e.g., CFI, TLI, RMSEA, SRMR, see Putnick & Bornstein, 2016)
  - Non-invariant measurement: tests **partial** measurement invariance

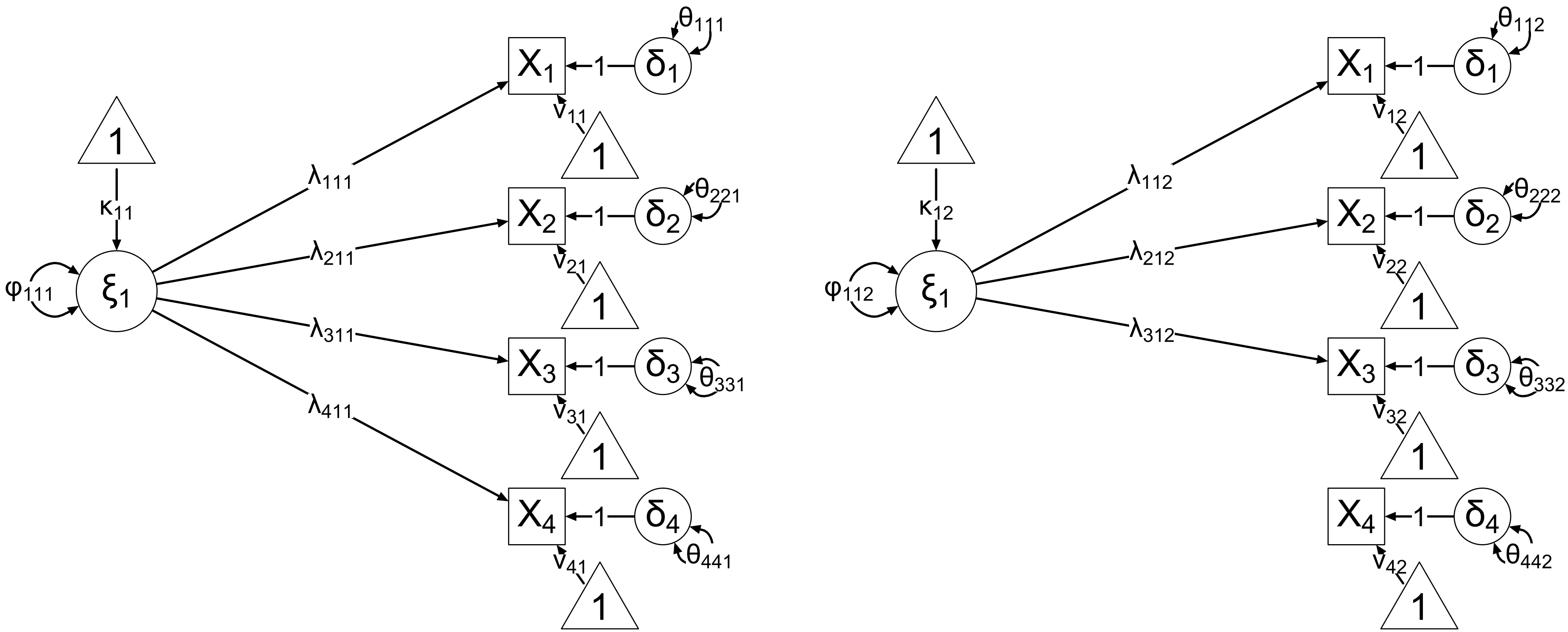
Measurement Invariance

- **Levels of measurement invariance:**
  - **Configural:** loading pattern is the same across groups
  - **Metric (weak):** respective loadings are the same across groups
  - **Scalar (strong):** respective intercepts are the same across groups
  - **Strict:** respective residual variances are the same across groups

Measurement Non-Invariance

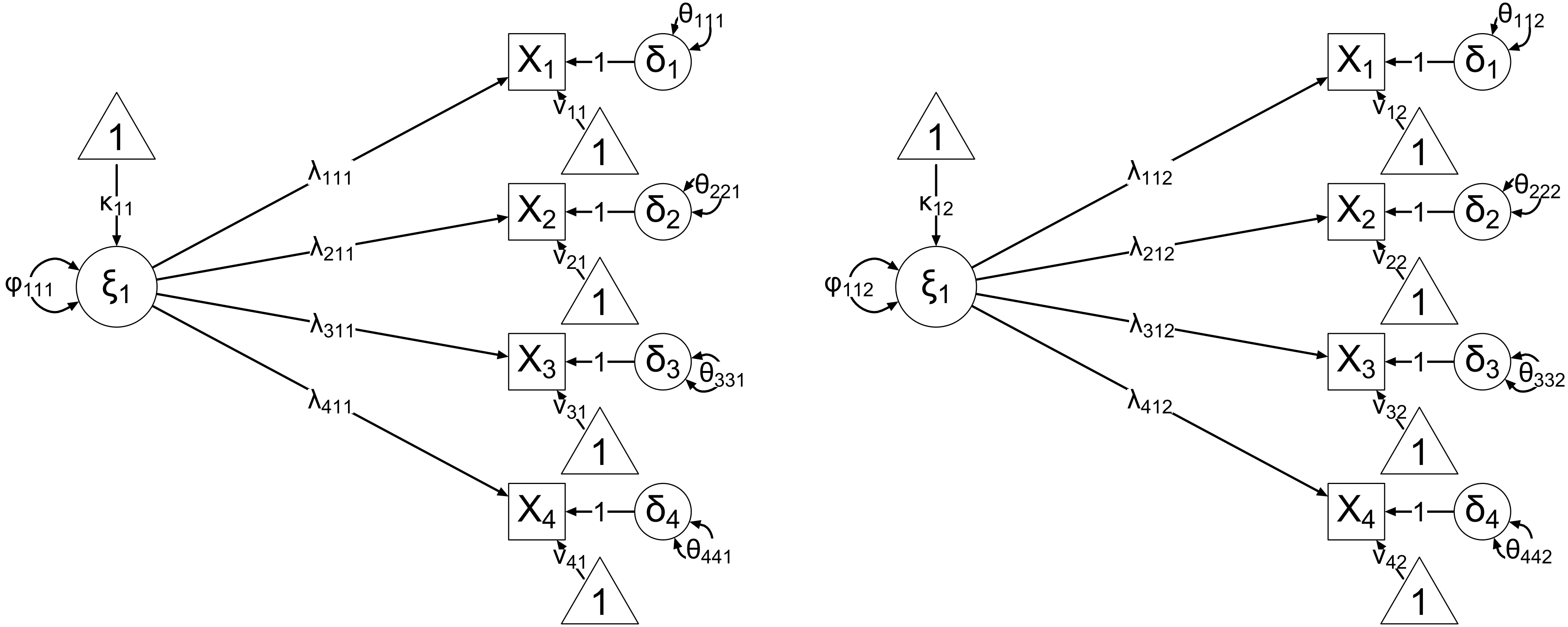
- **Consequences** of measurement non-invariance (Guenole & Brown, 2014):
  - **Configural:** different constructs, i.e., no valid comparisons across groups
  - **Metric (weak):** comparisons of **relationships** between **latent** variables (e.g., regression paths) across groups are invalid
  - **Scalar (strong):** comparisons of **latent means** across groups are biased and potentially misleading
  - **Strict: differences** in **observed** scores may reflect differences in measurement error rather than differences in the latent construct
    - Not a prerequiste for value comparison (reliabilities can differ across M)
- **Solution** to measurement non-invariance: **Partial MI**
  - Steenkamp & Baumgartner (1998) suggested that it may be sufficient to guarantee that only two items display equal factor loadings and intercepts
  - Recent simulations by Pokropek et al. (2019) concluded that it performs well

Configural Measurement Variance



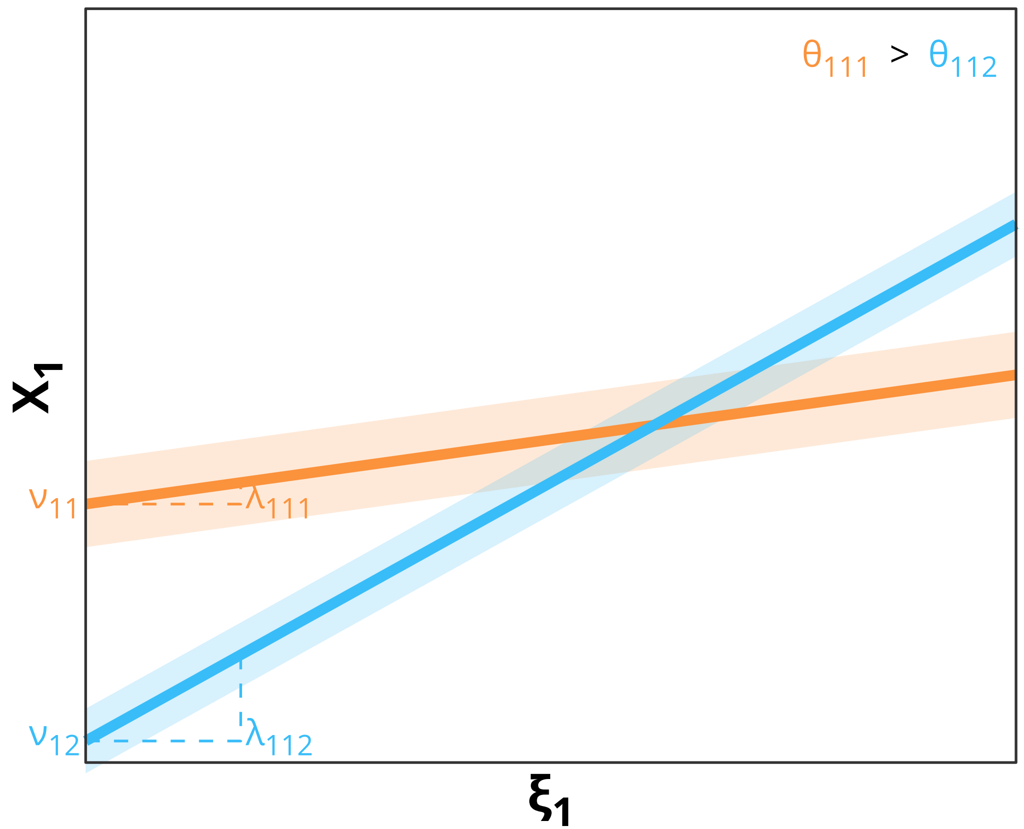
Different Loading Patterns Across Groups

Configural Measurement Invariance

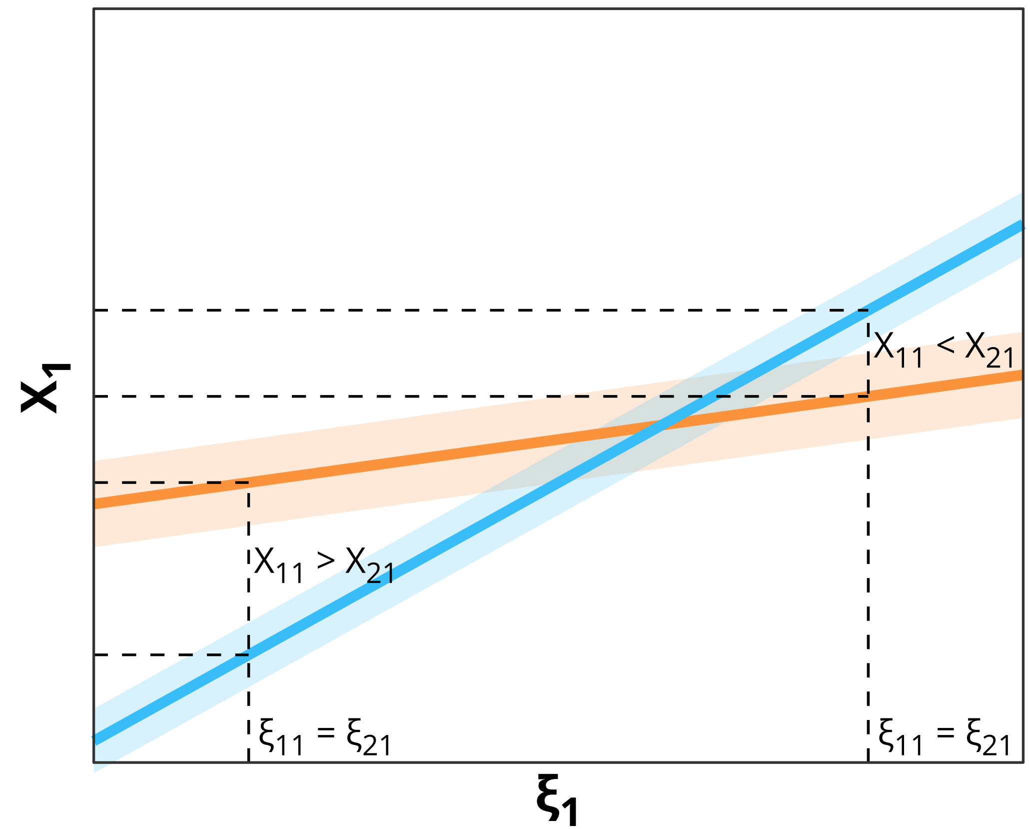


Equal Loading Patterns Across Groups

Configural Measurement Invariance

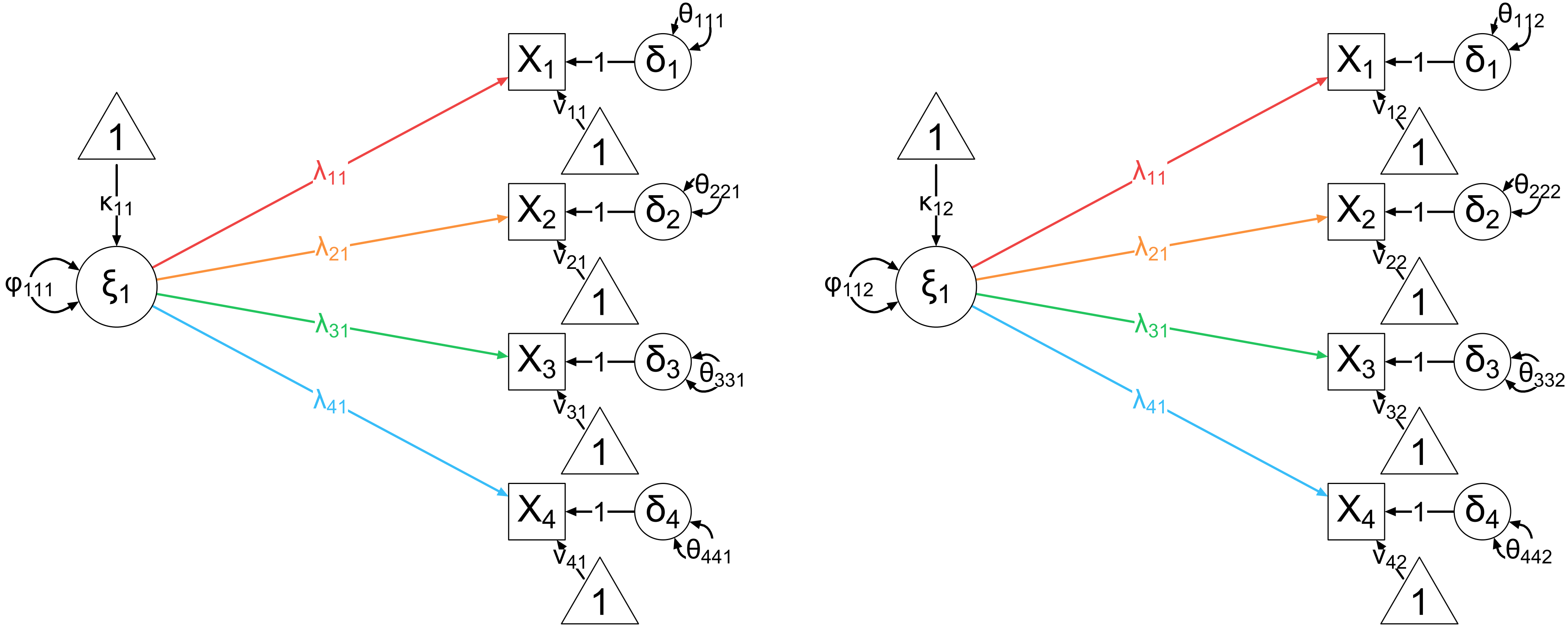


Different Loadings, Intercepts, and Variances Estimates Across Groups



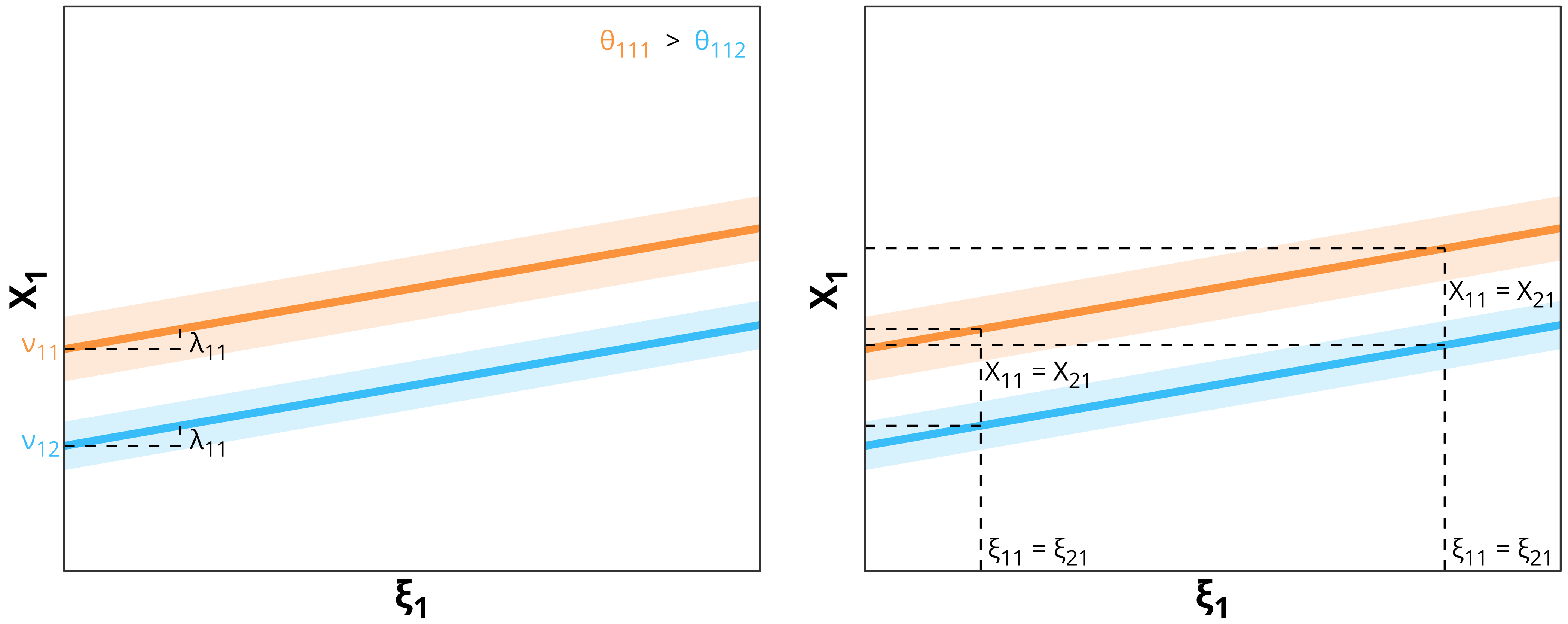
Consequence for Item Score Comparisons on Local Factor Score Levels

Metric Measurement Invariance



Equal Loadings Across Groups

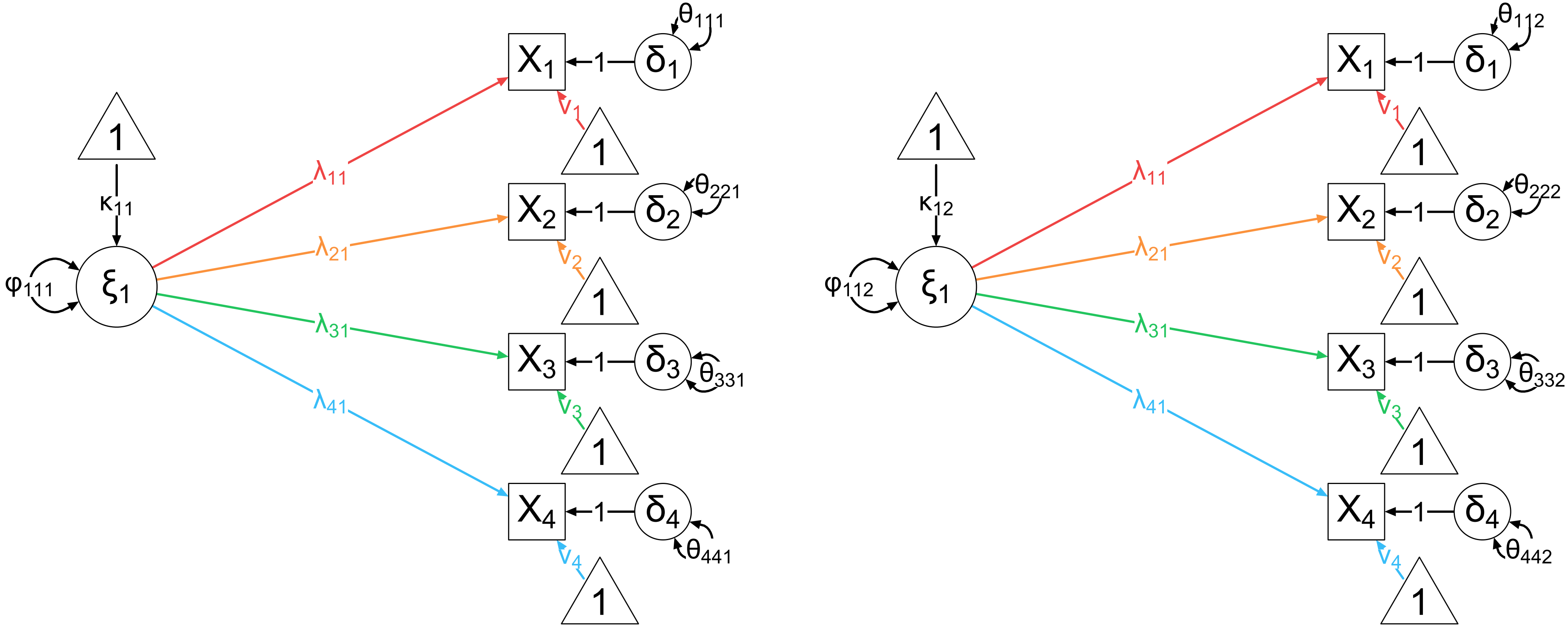
Metric Measurement Invariance



Equal Loadings, but Different Intercepts and Variances Across Groups

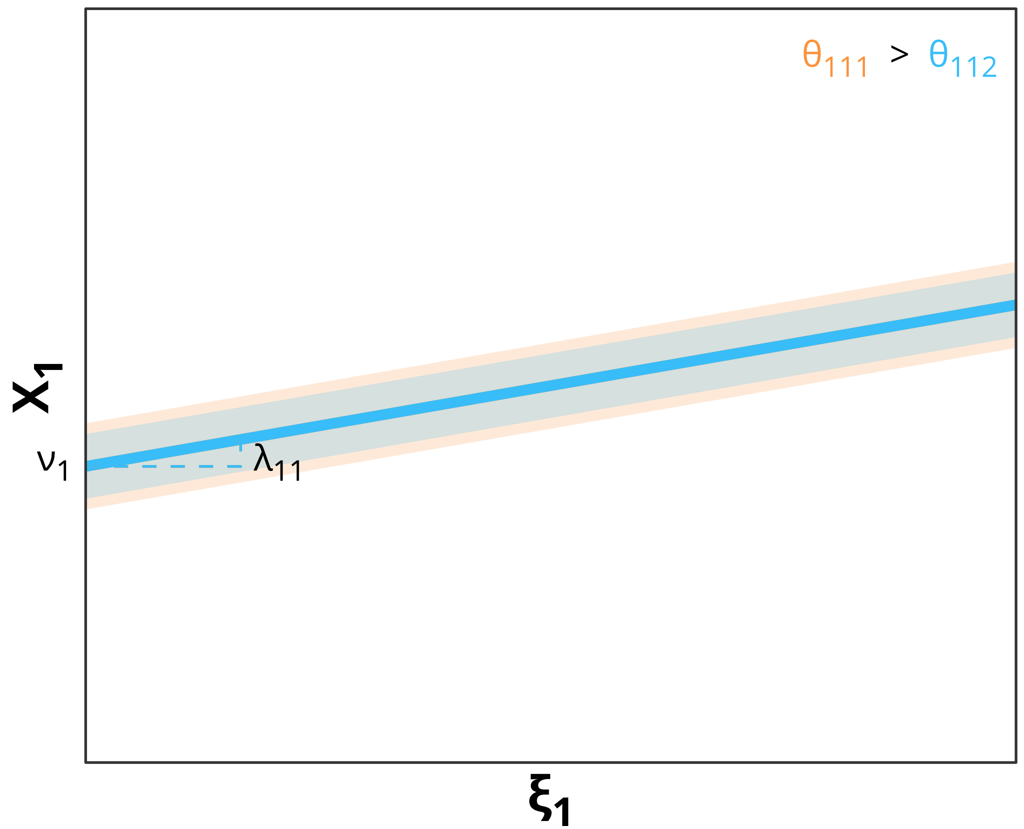
Consequence for Item Score Comparisons on Local Factor Score Levels

Scalar Measurement Invariance

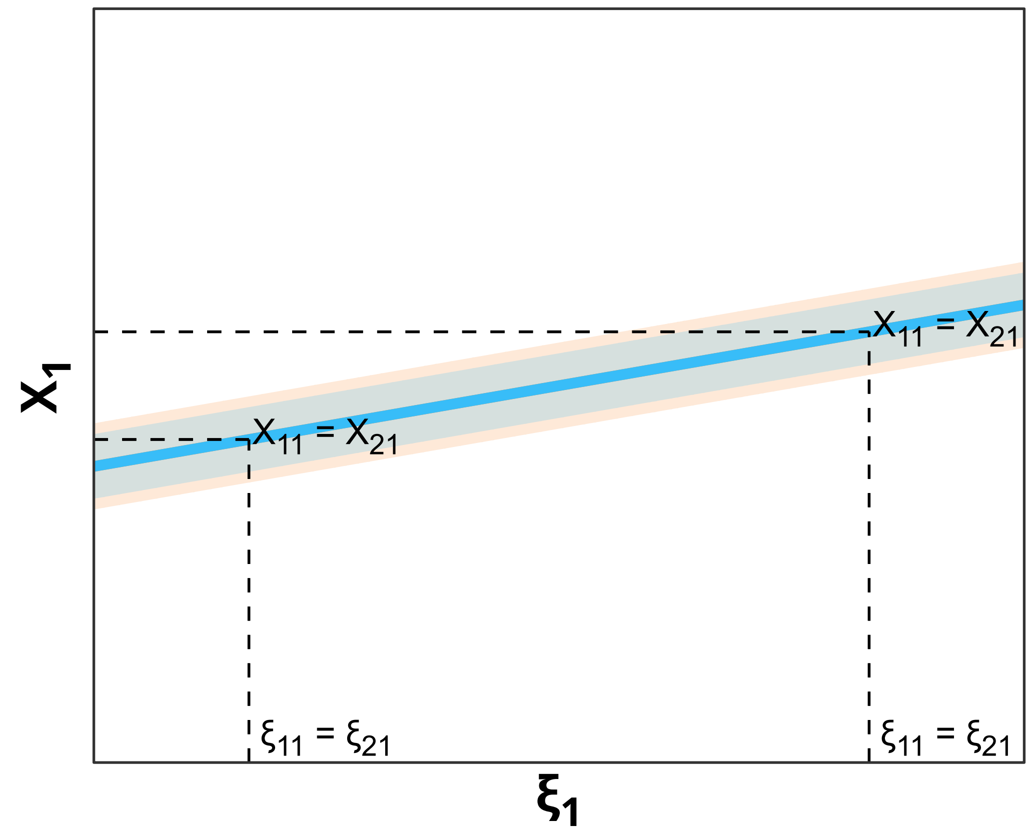


Equal Loadings and Intercepts Across Groups

Scalar Measurement Invariance

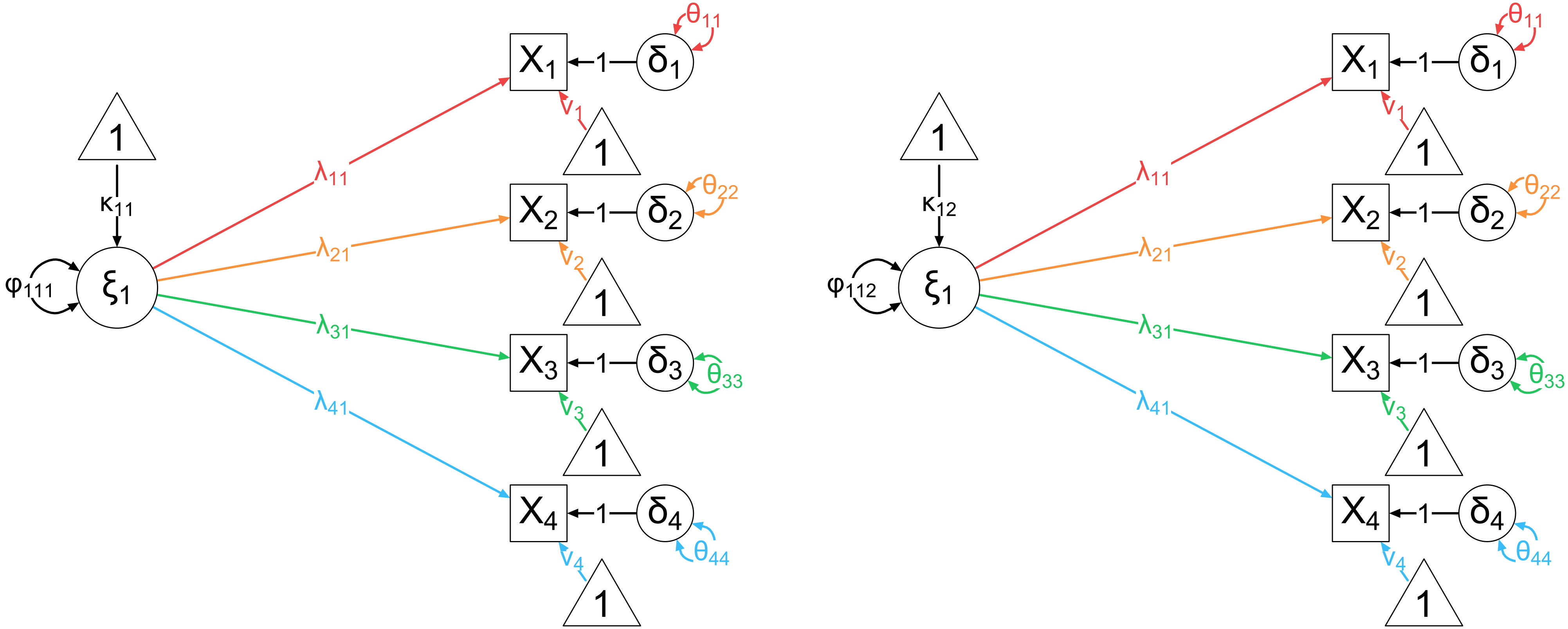


Equal Loadings and Intercepts, but Different Variances Across Groups



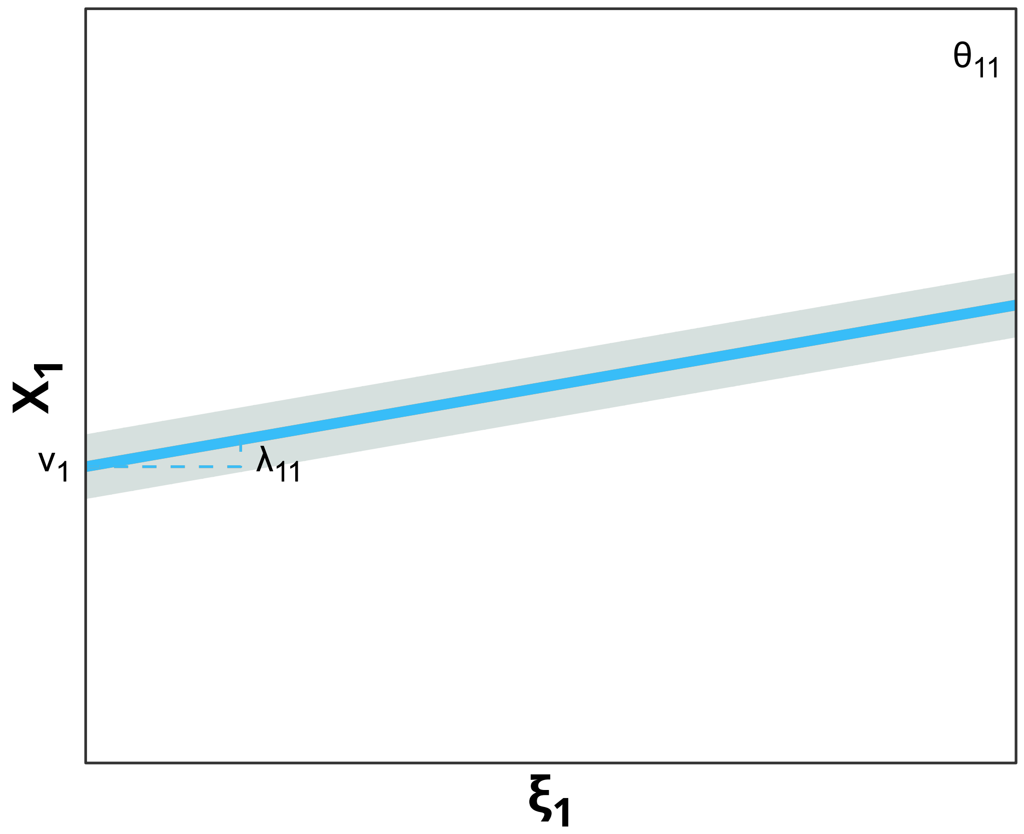
Consequence for Item Score Comparisons on Local Factor Score Levels

Strict Measurement Invariance

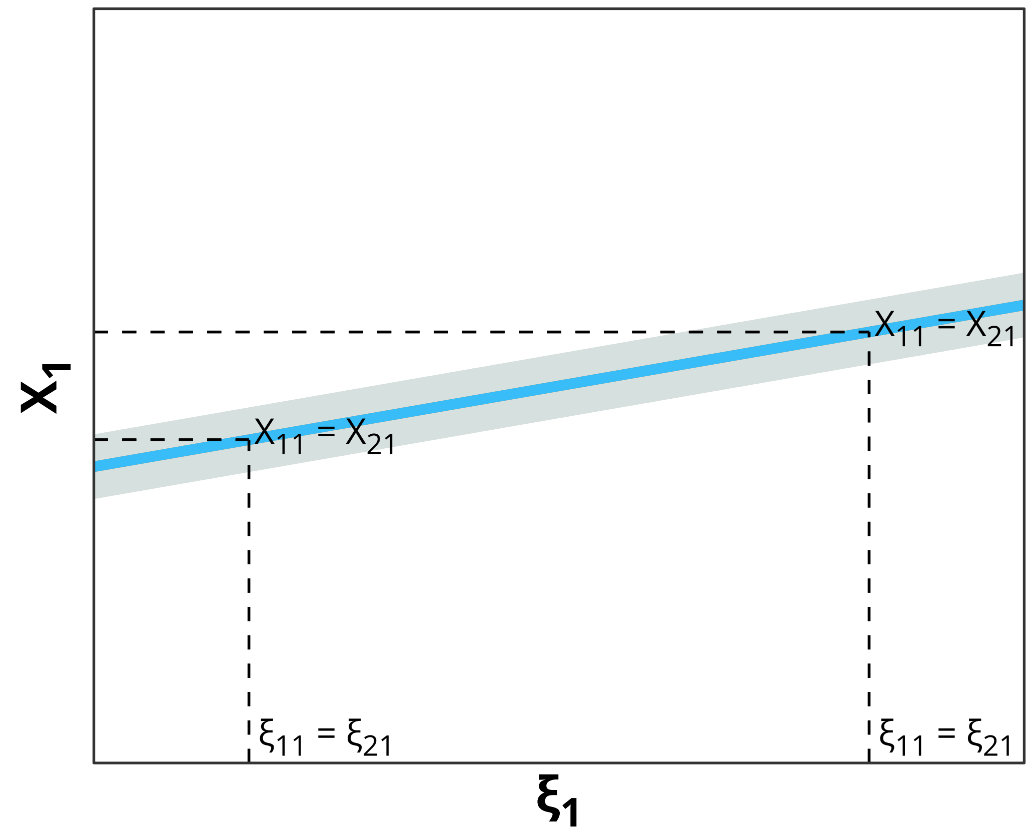


Equal Loadings, Intercepts, and Variances Across Groups

Strict Measurement Invariance



Equal Loadings, Intercepts, and Variances Across Groups



Consequence for Item Score Comparisons on Local Factor Score Levels

Continuous Variables

Level	Factor loadings	Intercepts	Residual variances	Factor means
Configural	*	*	*	Fixed at 0
Weak	Fixed	*	*	Fixed at 0
Strong	Fixed	Fixed	*	Fixed at 0/*
Strict	Fixed	Fixed	Fixed	Fixed at 0/*

- Parameter constraints according to Schroeders & Wilhelm (2011):
  - * = freely estimated
  - Fixed = fixed to equity across groups
  - Fixed at 0 = fixed at 0 in both groups
  - Fixed at 0/* = fixed at 0 in one group and freely estimated in the other

Categorical Variables

Level	Factor loadings	Thresholds	Residual variances	Factor means
Configural	(*	*)	Fixed at 1	Fixed at 0
Strong	(Fixed	Fixed)	Fixed at 1/*	Fixed at 0/*
Strict	(Fixed	Fixed)	Fixed at 1	Fixed at 0/*

- Parameter constraints according to Schroeders & Wilhelm (2011):
  - * = freely estimated
  - Fixed = fixed to equity across groups
  - Fixed at {.} = fixed at {.} in both groups
  - Fixed at {.}/* = fixed at {.} in one group and freely estimated in the other
  - ({.1} | {.2}) = parameters {.1} and {.2} need to be varied in tandem

Session 3

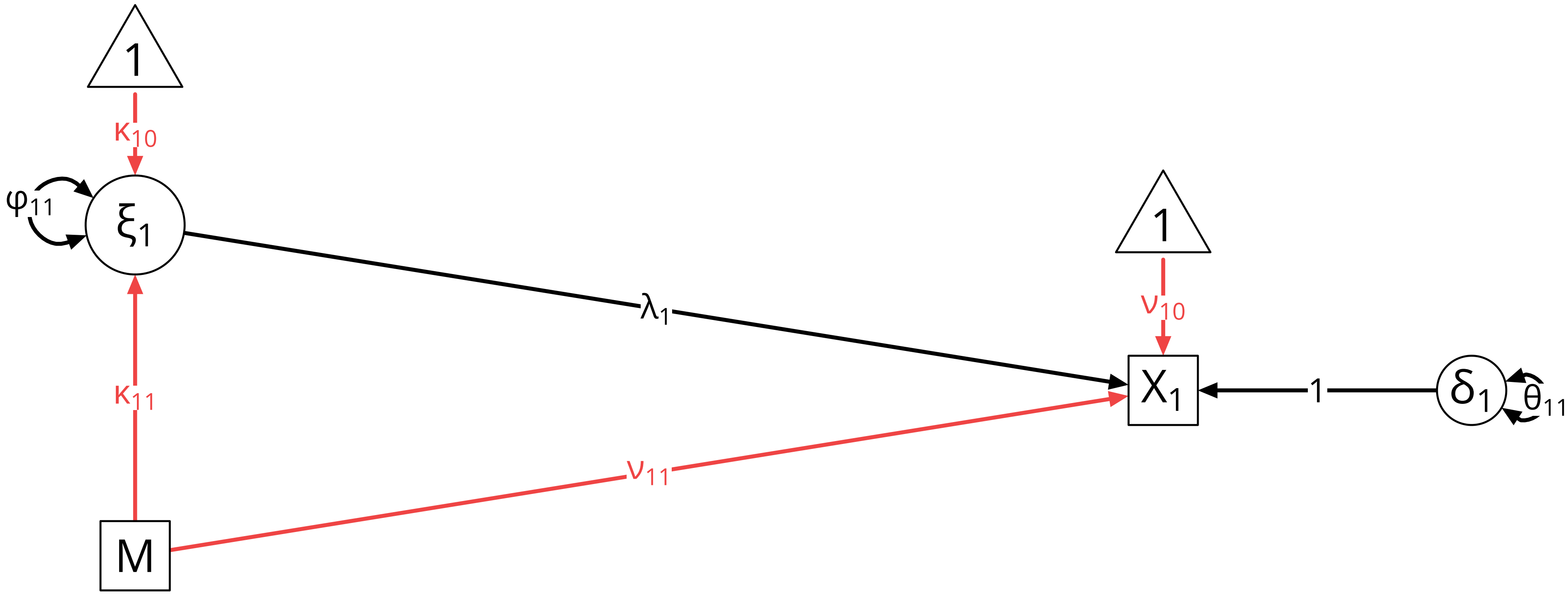
Moderated Nonlinear SEM

14:30–15:30

Moderated SEM

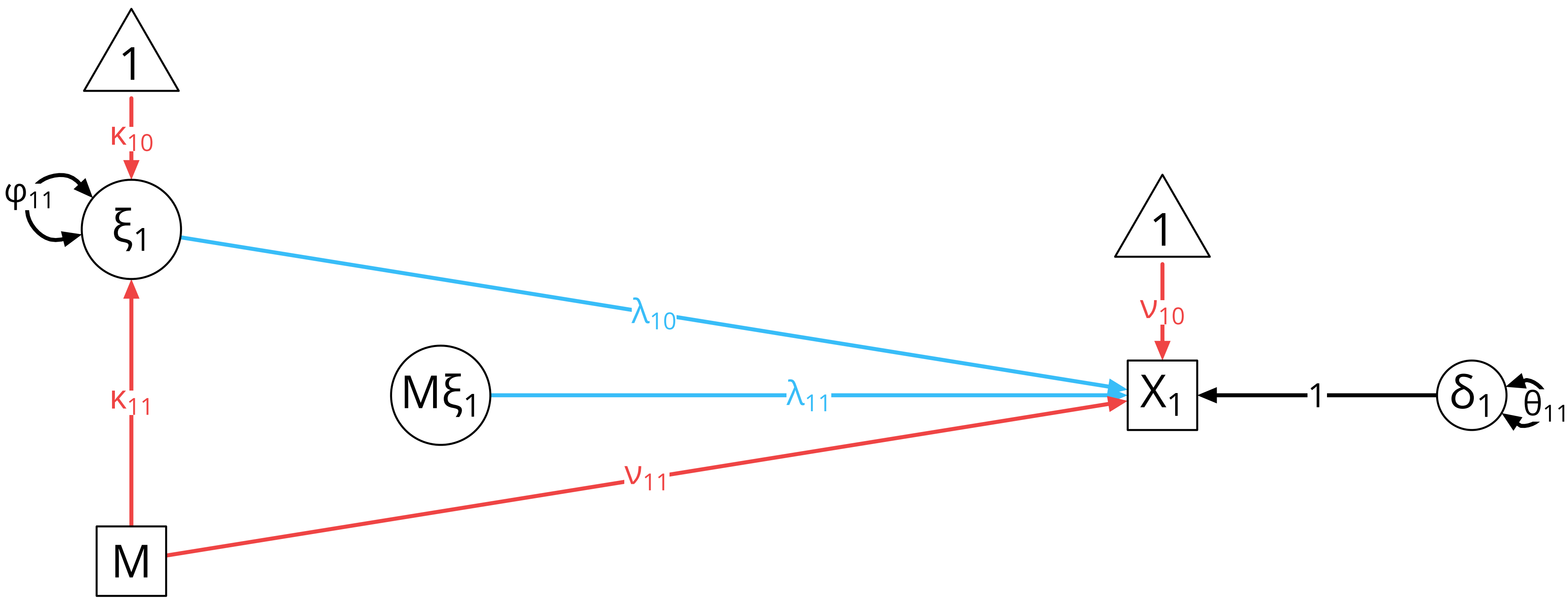
- **Basic idea:** examine **factor model parameters** (i.e., loadings, intercepts, residual variances) as deterministic **parametric functions** of **observed** continuous (or categorical) **covariates** (e.g., age)
- **Model overview** (Bauer, 2017):
  - **Multiple Group** (MG): only **categorical** effects on **all** parameters
  - **Multiple-Indicators-Multiple Causes** (MIMIC): categorial **+ continuous** parameters
    - **MIMIC:** effects on **means** and **intercepts**
    - **MIMIC+Interaction:** MIMIC parameters + **loadings**
  - **Moderated (Nonlinear) Factor Analysis** (MNLFA, Bauer & Hussong, 2009; Curran et al., 2014): categorial **+ continuous** parameters
    - **MNLFA:** MIMIC+Interaction parameters + **(residual) variances** (all parameters)

MIMIC Models



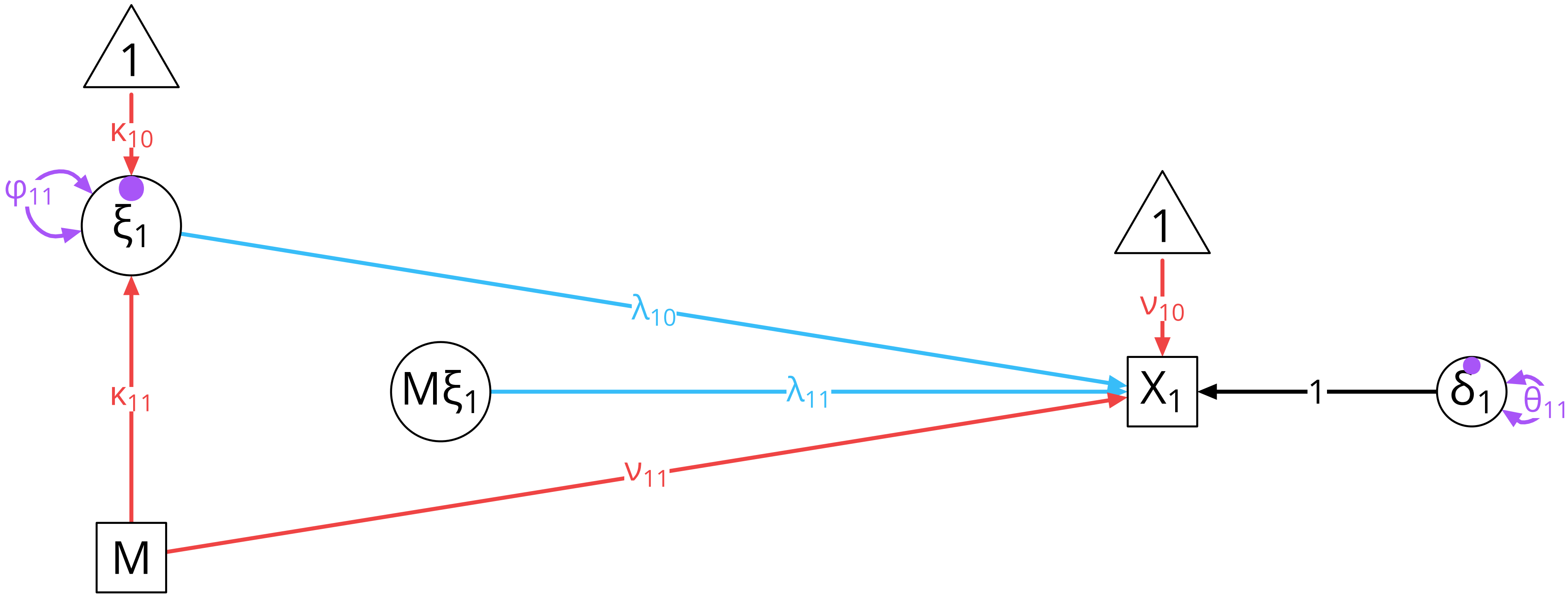
Multiple-Indicators-Multiple-Causes (MIMIC) Model (Only 1 Indicator Shown)

MIMIC-Interaction Models



Multiple-Indicators-Multiple-Causes (MIMIC) Model with Interaction Effect (Only 1 Indicator Shown)

Moderated Nonlinear Factor Analysis



Moderated Nonlinear Factor Analysis (Only 1 Indicator Shown; $\varphi_{11}(M)$ and $\theta_{11}(M)$ Could Additionally be Modeled as Effect Parameters)

Moderated Nonlinear Factor Analysis

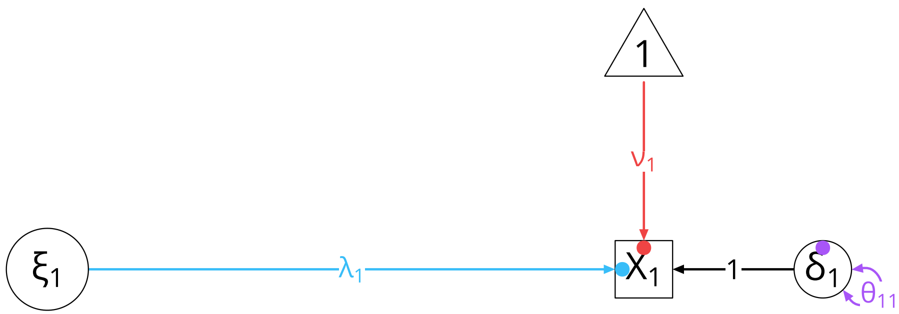
	Model			
Parameters	**MG**	**MIMIC**	**MIMIC-Interaction**	**MNLFA**
α, κ	G	G+C	G+C	G+C
φ, ψ	G	-	-	G+C
ν	G	G+C	G+C	G+C
λ	G	-	G+C	G+C
θ	G	-	-	G+C

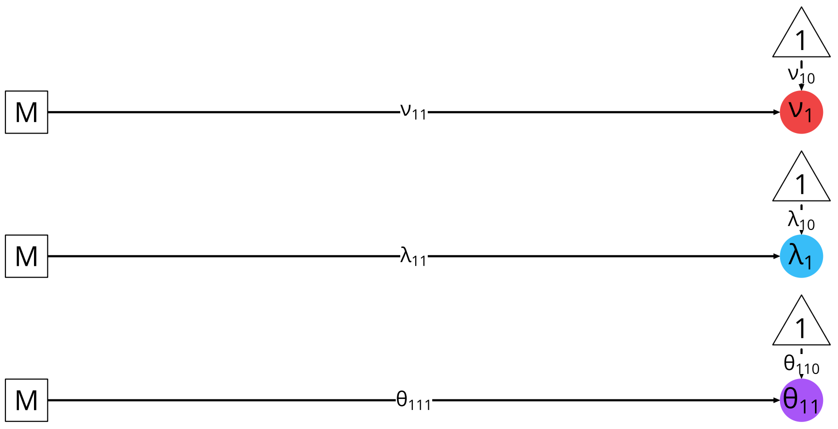
- Overview of different models by Bauer (2017) (slightly modified):
  - G = only grouping, i.e., categorical M
  - G+C = both categorical and continuous M

Latent Varying Coefficient Model

- **MNLFA:** Moderated latent regression with varying coefficients:

$$X_1 = \nu_1(M) + \lambda_1(M)\xi_1 + \delta_1, \delta_1 \sim \mathcal{N}(0, \theta^{\delta}_{11}(M))$$





Moderated Nonlinear Factor Analysis

- **Illustrative model:**
  - **Factor model:** $X_i = \nu_i(M) + \lambda_{ij}(M)\xi_j + \varepsilon_i, \delta_i \sim \mathcal{N}(0, \theta^{\delta}_{ii}(M))$
    - **Example**: linear and quadratic age effects for factor loading: $\lambda_{ij}(M) = \lambda_{ij0} + \lambda_{ij1}M + \lambda_{ij2}M^2$
- **Parameter interpretation:**
  - **Intercepts**: expected values at M = 0 (when centered: the mean)
  - **Linear and quadratic coefficients**: describe parameters change across the M (age)

Moderated Nonlinear Factor Analysis

- All coefficients of the measurement model could be moderated (here: only linear effect):

$$X_1 = \nu_1(M) + \lambda_1(M)\xi_1 + \delta_1, \delta_1 \sim \mathcal{N}(0, \theta^{\delta}_{11}(M))$$

$$\nu_1(M) = \nu_{10} + \nu_{11}M + \nu_{12}M^2$$

$$\lambda_1(M) = \lambda_{10} + \lambda_{11}M + \lambda_{12}M^2$$

$$\theta^{\delta}_{11}(M) = e^{\theta^{\delta}_{110} + \theta^{\delta}_{111}M + \theta^{\delta}_{112}M^2}$$

Moderated Nonlinear Factor Analysis

- **Fields of application:** testing measurement invariance across continuous moderator variables
  - **Special cases:**
    - Grouping variables (i.e., categorical M) → Multiple Group SEM
    - Interactions between grouping variables
    - Interactions between grouping and continous variables (e.g., Kolbe et al., 2022)
- **Extension:** Moderated **Nonlinear** SEM (MNSEM or MNLSEM)

Moderated Nonlinear SEM

- **Focus of this workshop:** application of the parametric varying coefficient framework to model the **nonlinear** effects of one (or more) **manifest moderators** M on the **coefficients** of the **structural model** of SEM, i.e., interaction of a latent variable with a manifest variable
  - **Focus:** structural model
    - $\eta_i = \alpha_i(M) + \gamma_{ij}(M)\xi_j + \zeta_i, \zeta_i \sim \mathcal{N}(0, \psi_{ii}(M))$
  - **Excursus:** measurement model (measurement invariance)
    - $X_k = \nu_k^{(x)}(M) + \lambda_{kj}^{(x)}(M)\xi_j + \delta_i, \delta_i \sim \mathcal{N}(0, \theta_{kk}^{\delta}(M))$
    - $Y_k = \nu_k^{(y)}(M) + \lambda_{ki}^{(x)}(M)\eta_i + \varepsilon_i, \varepsilon_i \sim \mathcal{N}(0, \theta_{kk}^{\varepsilon}(M))$

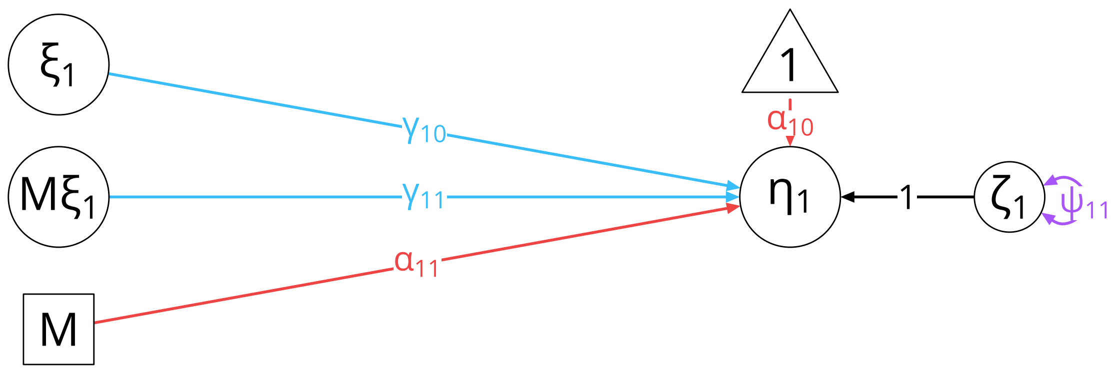
Latent Moderated Structures (LMS)

We will not discuss the effects of a latent moderator μ on coefficients in this workshop, i.e., interactions of latent variables.

Moderated Nonlinear SEM

- **MIMIC-Interaction:** Moderated regression with 1 latent and 1 observed predictor:

$$\eta_1 = \alpha_{10} + \gamma_{10}\xi_1 + \alpha_{11}M + \gamma_{11}\xi_1 M + \zeta_1, \zeta_1 \sim \mathcal{N}(0, \psi_{11})$$



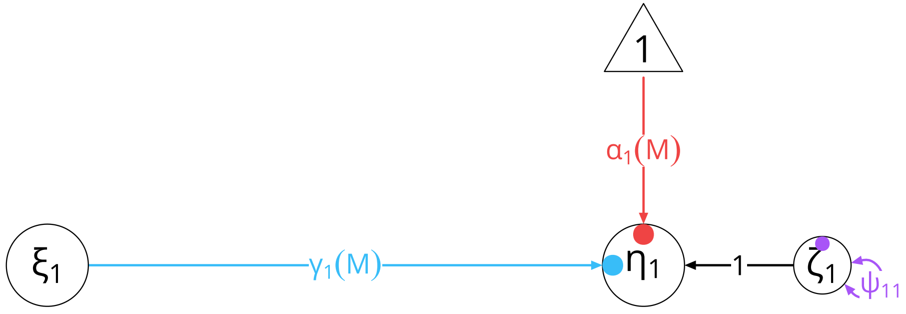
Moderated Latent Regression (With Focus on Effects on η_1)

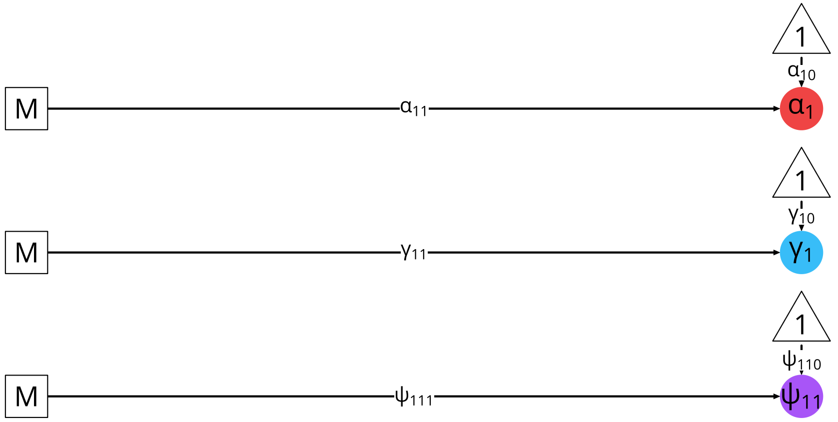
- **Coefficients:**
  - α_{10} = intercept, i.e., baseline of η_1
  - α_{11} = effect of M on the intercept of η_1
  - γ_{10} = baseline (linear) effect of ξ_1
  - γ_{11} = linear effect of M on the effect of ξ_1 on η_1

Latent Varying Coefficient Model

- **MNLFA:** models effects on residual variances (here: by an exponential link, will be explained later):

$$\eta_1 = \alpha_1(M) + \gamma_1(M)X + \zeta_1, \zeta_1 \sim \mathcal{N}(0, \psi_{11}(M))$$
$$\alpha_1(M) = \alpha_{10} + \alpha_{11}M$$
$$\gamma_1(M) = \gamma_{10} + \gamma_{11}M$$
$$\psi_{11}(M) = e^{\psi_{110} + \psi_{111}M}$$





Software Solutions

- **`lavaan`** itself does unfortunately ship support for varying coefficients
- **Other options:**
  - **`Mplus`**: extremely powerful, but proprietary
  - **`OpenMx`**: powerful and open source, but steep learning curve and intricate documentation (e.g., only scattered blog discussions on advanced analysis techniques)
    - **`mxsem`**: wrapper around `OpenMx` that facilitates building MNSEMs
  - **`Stan`** or **`JAGS`** (or `Mplus`): Bayesian modeling

`Mplus`

- Swiss army knife for all flavors SEM
- **Capabilities:** multilevel SEM, mixture modeling, complex survey data analysis, Bayesian analysis, Monte Carlo simulation
- **Model definition:** equations-based notation (comparable to `lavaan`s approach)
  - **Advantage:** great flexibility due to object-oriented nature
  - **Disadvantage:** proprietary (ca. € 900 per person, and personal opinion: outdated IDE)

Mplus Syntax

```
TITLE: Moderated Nonlinear Regression Model

DATA:
FILE = "data/performance.dat";

VARIABLE:
NAMES = aca_per stu_hou sle_qua stu_hou_c sle_qua_c;
USEVARIABLES = aca_per stu_hou_c;
CONSTRAINT = sle_qua_c;

MODEL:
[aca_per] (b0);
aca_per ON stu_hou_c (b1);

MODEL CONSTRAINT:
NEW (b00 b01 b02 b10 b11 b12);
b0 = b00 + b01 * sle_qua_c + b02 * sle_qua_c^2;
b1 = b10 + b11 * sle_qua_c + b12 * sle_qua_c^2;

OUTPUT: CINTERVAL TECH1;
```

OpenMx

- R rewrite of Mx (matrix processor and numerical optimizer) by Boker et al. (2011)
- **Capabilities:** e.g., factor mixture models, latent class models, multivariate ordinal models, genetic epidemiological models
- **Model definition:** equations-based notation or matrix-based notation (or combination)
  - **Advantage:** great flexibility due to object-oriented nature
  - **Disadvantage:** steep learning curve due to complex syntax

OpenMx Syntax

Path notation (“stepwise style”):

```
library(OpenMx)
data(demoOneFactor)
manifests <- names(demoOneFactor)
latents <- c("xi1")

factorModel1 <- mxModel(
    name = "One Factor",
    type = "RAM",                                                                          ①
    manifestVars = manifests,                                                              ②
    latentVars = latents,                                                                  ③
    mxPath(from = latents, to = manifests),                                                ④
    mxPath(from = manifests, arrows = 2),                                                  ⑤
    mxPath(from = latents, arrows = 2, free = FALSE, values = 1.0),                        ⑥
    mxData(observed = cov(demoOneFactor), type = "cov", numObs = 500)                      ⑦
)

factorFit1 <- mxRun(factorModel1)
summary(factorFit1)
```

OpenMx Syntax

Matrix notation (“classic style”):

```
library(OpenMx)
data(demoOneFactor)

factorModel2 <- mxModel(
    name = "One Factor",
    mxMatrix(type = "Full", nrow = 5, ncol = 1, free = TRUE, values = 0.2, name = "facLoadings"),  (1)
    mxMatrix(type = "Diag", nrow = 5, ncol = 5, free = TRUE, values = 1, name = "resVariances"),  (2)
    mxMatrix(type = "Symm", nrow = 1, ncol = 1, free = FALSE, values = 1, name = "facVariances"),  (3)
    mxAlgebra(expression = facLoadings %*% facVariances %*% t(facLoadings) + resVariances, name = "expCov"),  (4)
    mxExpectationNormal(covariance = "expCov", dimnames = names(demoOneFactor)),  (5)
    mxFitFunctionML(),  (6)
    mxData(observed = cov(demoOneFactor), type = "cov", numObs = 500)
)

factorFit2 <- mxRun(factorModel2)
summary(factorFit2)
```

`mxsem`

- `mxsem` (Orzek, 2024): `R` package providing `lavaan`-like syntax to implement SEMs with `OpenMx` (Boker et al., 2011)
- **Advantages:**
  - Facilitates model building due to user-friendly `lavaan`-like syntax **and** …
    - … adds capabilities for **parameter transformation** and definition variables
      - This is what we need for moderated nonlinear SEM
    - … ships extensibility of `OpenMx`
- **Disadvantage:** slightly different syntax from `lavaan`, global fit indices not reliable (inherited from `OpenMx`)
  - **Problem:** no agreed procedure on how to deal with definition variables in independent and saturated models
- **Alternatives:** metaSEM, umx, tidySEM, ezMx

`mxsem` Syntax

Definitions:

Formula type	Operator	Meaning
latent variable definition	`=~`	is measured by
regression	`~`	is regressed on
(residual) covariance	`~~`	is correlated with
intercept	`~ 1`	intercept
constraints / definitions	`:=`	is defined by
initialization	`!`	will be defined
covariates	`data.`	value can be found in the dataset

Moderated Nonlinear Regression

- `lm()` syntax:

```r
fit_lm <- lm(
    aca_per ~ 1 + sle_qua_c + I(sle_qua_c^2) +
        stu_hou_c + sle_qua_c:stu_hou_c + I(sle_qua_c^2):stu_hou_c,
    data = performance
)
```

- `mxsem()` syntax:

```r
model <- "
  # Path model
  aca_per ~ b0 * 1 + b1 * stu_hou_c

  # Create new parameters
  !b00; !b01; !b02
  !b10; !b11; !b12

  # Redefine b0 and b1
  b0 := b00 + b01 * data.sle_qua_c + b02 * data.sle_qua_c^2
  b1 := b10 + b11 * data.sle_qua_c + b12 * data.sle_qua_c^2
"

model_mx <- mxsem(model, data = performance)
fit_mx <- mxRun(model_mx)
```

`mxsem` Model Syntax

- Explicit definition, i.e., with initialization `!`:

```
model <- "
  # Path model
  aca_per ~ b0 * 1 + b1 * stu_hou_c

  # Initialization: Create new parameters
  !b00; !b01; !b02
  !b10; !b11; !b12

  # Redefine b0 and b1
  b0 := b00 + b01 * data.sle_qua_c + b02 * data.sle_qua_c^2
  b1 := b10 + b11 * data.sle_qua_c + b12 * data.sle_qua_c^2
"
```

- Implicit definition, i.e., inline, with curly braces `{}`:

```
model <- "
  # Moderation on the intercept
  aca_per ~ {b0 := b00 + b01 * data.sle_qua_c + b02 * data.sle_qua_c^2} * 1
  # Moderation on the slope
  aca_per ~ {b1 := b10 + b11 * data.sle_qua_c + b12 * data.sle_qua_c^2} * stu_hou_c
"
```

Excursus: OpenMx Code

Just to get a feeling about what is handled by `mxsem`:

```
library(OpenMx)

dims <- list(c("aca_per", "stu_hou_c"), c("aca_per", "stu_hou_c"))

matrix_A <- mxMatrix(                                                          ①
    type = "Full", nrow = 2, ncol = 2, free = FALSE,
    values = 0, name = "A", labels = c(NA, NA, "b1[1,1]", NA), dimnames = dims
)
matrix_S <- mxMatrix(                                                          ②
    type = "Symm", nrow = 2, ncol = 2, free = c(TRUE, FALSE, FALSE, TRUE),
    values = c(.1, 0, 0, .1), lbound = c(1e-6, NA, NA, 1e-6), name = "S",
    labels = c("aca_per↔aca_per", NA, NA, "stu_hou_c↔stu_hou_c"), dimnames = dims
)
matrix_F <- mxMatrix(                                                          ③
    type = "Full", nrow = 2, ncol = 2, free = FALSE,
    values = c(1, 0, 0, 1), name = "F", dimnames = dims
)
matrix_M <- mxMatrix(                                                          ④
    type = "Full", nrow = 1, ncol = 2, free = c(FALSE, TRUE),
    values = c(0, 0), name = "M", labels = c("b0[1,1]", "one→stu_hou_c"),
    dimnames = list(NULL, dims[[1]])
)
matrix_NP <- mxMatrix(                                                         ⑤
    type = "Full", nrow = 1, ncol = 6, free = TRUE,
    values = 0.01, name = "new_parameters", labels = c(paste0("b0", 0:2), paste0("b1", 0:2))
)
```

Excursus: OpenMx Code

Just to get a feeling about what is handled by `mxsem`:

```r
algebra_b0 <- mxAlgebra(expression = b00 + b01 * data.sle_qua_c + b02 * data.sle_qua_c^2, name = "b0")
algebra_b1 <- mxAlgebra(expression = b10 + b11 * data.sle_qua_c + b12 * data.sle_qua_c^2, name = "b1")

model_omx <-
    mxModel(
        type = "RAM",
        manifestVars = dims[[1]],
        matrix_A, matrix_S, matrix_F, matrix_M, matrix_NP,
        algebra_b0, algebra_b1,
        mxExpectationRAM(M = "M"), mxFitFunctionML(),
        mxData(observed = performance, type = "raw")
    )

fit_omx <- mxRun(model_omx)
summary(fit_mx)
```

Moderated Nonlinear Path Analysis

- `lm()` output:

```
as_tibble(summary(fit_lm)$coefficients)
```

```
#> # A tibble: 6 × 4
#>   Estimate `Std. Error` `t value` `Pr(>|t|)`
#>      <dbl>        <dbl>     <dbl>      <dbl>
#> 1  42.4         0.681      62.2    6.65e-236
#> 2   0.517       0.157       3.30   1.03e-  3
#> 3  -0.0462      0.0602     -0.768  4.43e-  1
#> 4   0.547       0.120       4.54   7.01e-  6
#> 5   0.302       0.0272     11.1    8.13e- 26
#> 6  -0.151       0.0103    -14.7    1.00e- 40
```

- `mxsem()` output:

```
as_tibble(summary(fit_mx)$parameters[c("name", "matrix", "Estimate", "Std.Error")])
```

```
#> # A tibble: 9 × 4
#>   name                 matrix             Estimate Std.Error
#>   <chr>                <chr>                 <dbl>     <dbl>
#> 1 aca_per↔aca_per      S              102.             6.42
#> 2 stu_hou_c↔stu_hou_c  S               32.3            2.04
#> 3 one→stu_hou_c        M                0.000000271    0.254
#> 4 b00                  new_parameters  42.4            0.677
#> 5 b01                  new_parameters   0.517          0.156
#> 6 b02                  new_parameters  -0.0462         0.0599
#> 7 b10                  new_parameters   0.547          0.120
#> 8 b11                  new_parameters   0.302          0.0270
#> 9 b12                  new_parameters  -0.151          0.0102
```

lavaan Syntax

Complete specification (without intercepts):

```
model <- "
  # Measurement model
  xi1 =~ x1 + x2 + x3                1
  eta1 =~ y1 + y2                    2

  x1 ~~ x1                           3
  x2 ~~ x2
  x3 ~~ x3
  y1 ~~ y1                           4
  y2 ~~ y2

  # Structural model
  eta1 ~ xi1                         5

  xi1 ~~ xi1                         6
  eta1 ~~ eta1
"
```

lavaan Syntax

Complete specification (with intercepts):

```
model <- "
  # Measurement model
  xi1 =~ x1 + x2 + x3                (1)
  eta1 =~ y1 + y2                    (2)

  x1 ~ 1                             (3)
  x2 ~ 1
  x3 ~ 1
  y1 + y2 ~ 1                        (4)

  x1 ~~ x1                           (5)
  x2 ~~ x2
  x3 ~~ x3
  y1 ~~ y1                           (6)
  y2 ~~ y2

  # Structural model
  eta1 ~ xi1                         (7)

  xi1 ~ 1                            (8)
  eta1 ~ 1

  xi1 ~~ xi1                         (9)
  eta1 ~~ eta1
"
```

lavaan Syntax

Complete specification with modifiers (freely customizable):

```
model <- "
  # Measurement model
  xi1 =~ lambda_x_11 * x1 + lambda_x_21 * x2 + lambda_x_31 * x3   (1)
  eta1 =~ lambda_y_11 * y1 + lambda_y_21 * y2   (2)

  x1 ~ nu_x_1 * 1   (3)
  x2 ~ nu_x_2 * 1
  x3 ~ nu_x_3 * 1
  y1 ~ nu_y_1 * 1   (4)
  y2 ~ nu_y_2 * 1

  x1 ~~ theta_x_11 * x1   (5)
  x2 ~~ theta_x_22 * x2
  x3 ~~ theta_x_33 * x3
  y1 ~~ theta_y_11 * y1   (6)
  y2 ~~ theta_y_22 * y2

  # Structural model
  eta1 ~ gamma_11 * xi1   (7)

  xi1 ~ kappa_1 * 1   (8)
  eta1 ~ alpha_1 * 1

  xi1 ~~ phi_11 * xi1   (9)
  eta1 ~~ psi_11 * eta1
"
```

Example: MNSEM

Example 2.2: Parent Depression and Internalizing in Adolescents (pure)

We are studying the **correlation** between latent variables of parent depression (Ds, ξ_1) and adolescents' internalizing problems (Is, ξ_2).

This correlation is influenced nonlinearly by adolescents' age (M) and we hypothesize that this effect is quadratic.

```r
load("data/pure.RData")  # (1)

library(mxsem)  # (2)
library(OpenMx)  # (3)

pure$c_cage <- pure$cage - mean(pure$cage)  # (4)
```

For the sake of simplicity, in our workshop, we assume scalar invariance of both latent variables across the different values of adolescent age (cage or, centered, c_cage), which is the minimum level required for comparing latent means and variances (Van de Schoot et al., 2012).

Example: MNSEM

```r
model_loadings <- "
  Depression =~ l_dep1 * pdep1 + l_dep2 * pdep2 + l_dep3 * pdep3 + l_dep4 * pdep4 +
                l_dep5 * pdep5 + l_dep6 * pdep6 + l_dep7 * pdep7 + l_dep8 * pdep8 +
                l_dep9 * pdep9

  Internalizing =~ l_int1 * cint1 + l_int2 * cint2 + l_int3 * cint3 +
                   l_int4 * cint4 + l_int5 * cint5
"

model_intercepts <- "
  pdep1 ~ i_dep1 * 1
  pdep2 ~ i_dep2 * 1
  pdep3 ~ i_dep3 * 1
  pdep4 ~ i_dep4 * 1
  pdep5 ~ i_dep5 * 1
  pdep6 ~ i_dep6 * 1
  pdep7 ~ i_dep7 * 1
  pdep8 ~ i_dep8 * 1
  pdep9 ~ i_dep9 * 1

  cint1 ~ i_int1 * 1
  cint2 ~ i_int2 * 1
  cint3 ~ i_int3 * 1
  cint4 ~ i_int4 * 1
  cint5 ~ i_int5 * 1
"
```

Bounding of Parameters

In SEM, different parameters have different range:

- Non-standardized **intercepts** and **factor loadings**: $[-\infty, +\infty]$
- **Variances**: $[0, +\infty]$
- **Correlations**: $[-1, 1]$

Estimation constraints to avoid improper estimates (Bauer, 2017):

- Variance parameters: **exponential** transformation
  - $\theta(M) = e^{\theta_0 + \theta_1 M + \theta_2 + M^2}$
- Correlations, we could use **Fisher's z-transformation**
  - $\rho(M) = \frac{e^{2\cdot(\rho_0 + \rho_1 M + \rho_2 + M^2)} - 1}{e^{2\cdot(\rho_0 + \rho_1 M + \rho_2 + M^2)} + 1}$

Example: MNSEM

```
model_covariances <- "
  pdep1 ~~ {v_dep1 := exp(v_dep1_0 + v_dep1_1 * data.c_cage + v_dep1_2 * data.c_cage^2)} * pdep1
  pdep2 ~~ {v_dep2 := exp(v_dep2_0 + v_dep2_1 * data.c_cage + v_dep2_2 * data.c_cage^2)} * pdep2
  pdep3 ~~ {v_dep3 := exp(v_dep3_0 + v_dep3_1 * data.c_cage + v_dep3_2 * data.c_cage^2)} * pdep3
  pdep4 ~~ {v_dep4 := exp(v_dep4_0 + v_dep4_1 * data.c_cage + v_dep4_2 * data.c_cage^2)} * pdep4
  pdep5 ~~ {v_dep5 := exp(v_dep5_0 + v_dep5_1 * data.c_cage + v_dep5_2 * data.c_cage^2)} * pdep5
  pdep6 ~~ {v_dep6 := exp(v_dep6_0 + v_dep6_1 * data.c_cage + v_dep6_2 * data.c_cage^2)} * pdep6
  pdep7 ~~ {v_dep7 := exp(v_dep7_0 + v_dep7_1 * data.c_cage + v_dep7_2 * data.c_cage^2)} * pdep7
  pdep8 ~~ {v_dep8 := exp(v_dep8_0 + v_dep8_1 * data.c_cage + v_dep8_2 * data.c_cage^2)} * pdep8
  pdep9 ~~ {v_dep9 := exp(v_dep9_0 + v_dep9_1 * data.c_cage + v_dep9_2 * data.c_cage^2)} * pdep9

  cint1 ~~ {v_int1 := exp(v_int1_0 + v_int1_1 * data.c_cage + v_int1_2 * data.c_cage^2)} * cint1
  cint2 ~~ {v_int2 := exp(v_int2_0 + v_int2_1 * data.c_cage + v_int2_2 * data.c_cage^2)} * cint2
  cint3 ~~ {v_int3 := exp(v_int3_0 + v_int3_1 * data.c_cage + v_int3_2 * data.c_cage^2)} * cint3
  cint4 ~~ {v_int4 := exp(v_int4_0 + v_int4_1 * data.c_cage + v_int4_2 * data.c_cage^2)} * cint4
  cint5 ~~ {v_int5 := exp(v_int5_0 + v_int5_1 * data.c_cage + v_int5_2 * data.c_cage^2)} * cint5
"
```

Example: MNSEM

```r
model_latent <- "
  Depression ~~ 1 * Depression
  Internalizing ~~ 1 * Internalizing

  Depression ~ 0 * 1
  Internalizing ~ 0 * 1

  Depression ~~ {cov := (exp(2*(cov_0 + cov_1 * data.c_cage + cov_2 * data.c_cage^2)) - 1) /
                        (exp(2*(cov_0 + cov_1 * data.c_cage + cov_2 * data.c_cage^2)) + 1)} * Internalizing
"
```

Fisher's Transformation

As the correlation ρ is a parameter bounded within -1 and 1, we need Fisher's transformation to accomodate for these bounds.

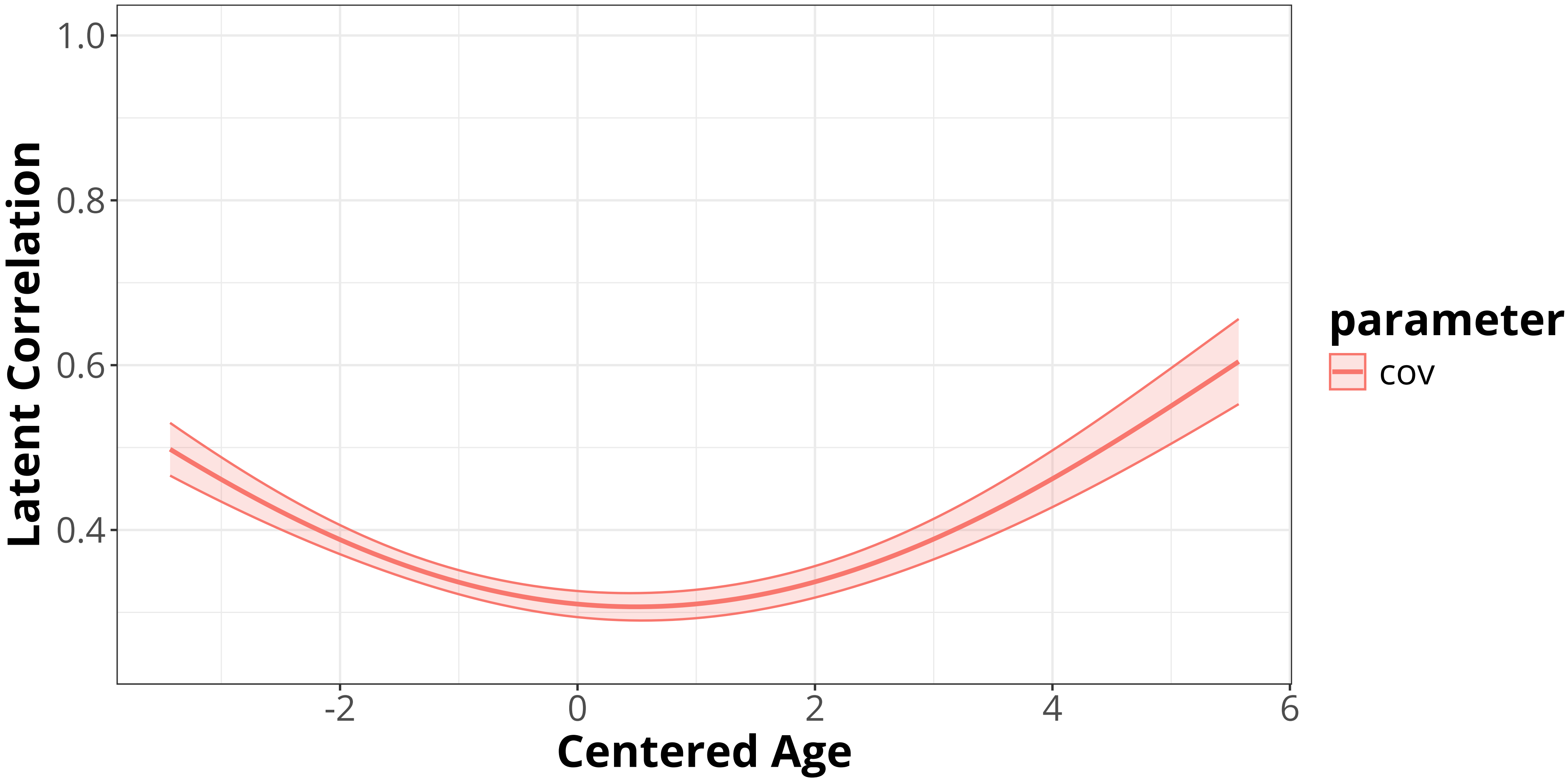
Example: MNSEM

```r
SEM_model <- paste0(model_loadings, model_intercepts,
    model_covariances, model_latent,
    collapse = "\n"
)

model_mnsem <-
    mxsem(
        model = SEM_model,
        data = pure,
        scale_loadings = FALSE,
        scale_latent_variances = FALSE
    )
fit_mnsem <- mxRun(model = model_mnsem)
```

- **Problem:** to my knowing, no easily available option to analytically derive confidence intervals of the cov parameter across the range of M
    - **Approach:** bootstrappig to estimate the bootrap SE

```r
boot_mnsem <- mxBootstrap(mxModel(fit_mnsem), replications = 100, intervals = TRUE)
summary(boot_mnsem)
```

MNSEM: Bootstrapping



Effects of the Moderator Variable Age on the Latent Correlation Coefficient

MNSEM: Parameter Tests

p-values can be computed based on the estimate and the bootstrap SE

```r
res_boot <- as_tibble(summary(boot_mnsem)$parameters[c("name", "matrix", "Estimate", "Std.Error")])
res_boot$Std.Error <- summary(boot_mnsem)$bootstrapSE
res_boot$p.value <- 2 * (1 - pnorm(abs(res_boot$Estimate / res_boot$Std.Error)))
filter(res_boot, name %in% c("cov_0", "cov_1", "cov_2"))
```

```
#> # A tibble: 3 × 5
#>   name  matrix         Estimate Std.Error  p.value
#>   <chr> <chr>             <dbl>     <dbl>    <dbl>
#> 1 cov_0 new_parameters   0.320    0.00896 0
#> 2 cov_1 new_parameters  -0.0146   0.00380 0.000120
#> 3 cov_2 new_parameters   0.0148   0.00148 0
```

MNSEM: Parameter Tests

- **Problem:** multiple tests → we should account for Type I error accumulation Kolbe et al. (2022)
  - **Solution:** Benjamini-Hochberg correction (e.g., Thissen et al., 2002)

```r
res_boot_new <- filter(res_boot, matrix == "new_parameters")
res_boot_new$p.adjusted <- p.adjust(res_boot_new$p.value, method = "BH")
filter(res_boot_new, name %in% c("cov_0", "cov_1", "cov_2"))
```

```
#> # A tibble: 3 × 6
#>   name  matrix         Estimate Std.Error  p.value p.adjusted
#>   <chr> <chr>             <dbl>     <dbl>    <dbl>      <dbl>
#> 1 cov_0 new_parameters   0.320    0.00896 0               0
#> 2 cov_1 new_parameters  -0.0146   0.00380 0.000120    0.000208
#> 3 cov_2 new_parameters   0.0148   0.00148 0               0
```

MNSEM: Model Comparison

- **Alternative test** for moderation of parameters: $\Delta\chi^2$-test
  - **Model 0:** restricted to have **no** moderation on parameter ρ
  - **Model 1:** restricted to have **linear** moderation on parameter ρ
  - **Model 2: quadratic** moderation on parameter $\rho(M)$

```r
model_latent <- "
  Depression ~~ 1 * Depression
  Internalizing ~~ 1 * Internalizing
  Depression ~ 0 * 1
  Internalizing ~ 0 * 1
"

model_latent_cor0 <- "
  Depression ~~ Internalizing
"

model_latent_cor1 <- "
  Depression ~~ {cov := (exp(2*(cov_0 + cov_1 * data.c_cage)) - 1) /
                        (exp(2*(cov_0 + cov_1 * data.c_cage)) + 1)} * Internalizing
"
SEM_model <- paste0(model_latent, model_loadings, model_intercepts, model_covariances, collapse = "\n")
SEM_model0 <- paste0(SEM_model, model_latent_cor0, collapse = "\n")
SEM_model1 <- paste0(SEM_model, model_latent_cor1, collapse = "\n")
```

MNSEM: Model Comparison

```r
model_mnsem0 <- mxsem(model = SEM_model0, data = pure, scale_loadings = F, scale_latent_variances = F)
model_mnsem1 <- mxsem(model = SEM_model1, data = pure, scale_loadings = F, scale_latent_variances = F)

fit_mnsem0 <- mxRun(model = model_mnsem0)
fit_mnsem1 <- mxRun(model = model_mnsem1)

mxCompare(fit_mnsem, comparison = list(fit_mnsem1, fit_mnsem0))
```

```
#>           base   comparison ep minus2LL     df      AIC   diffLL diffdf
#> 1 untitled5333         <NA> 73 374240.6 273823 374386.6       NA     NA
#> 2 untitled5333 untitled5589 72 374340.9 273824 374484.9 100.2205      1
#> 3 untitled5333 untitled5604 71 374342.8 273825 374484.8 102.1209      2
#>              p
#> 1           NA
#> 2 1.363415e-23
#> 3 6.679178e-23
```

As model fit gets worse, ρ should be modeled as a quadratic function of age.

Coffee Break

15:30–16:00

Session 4

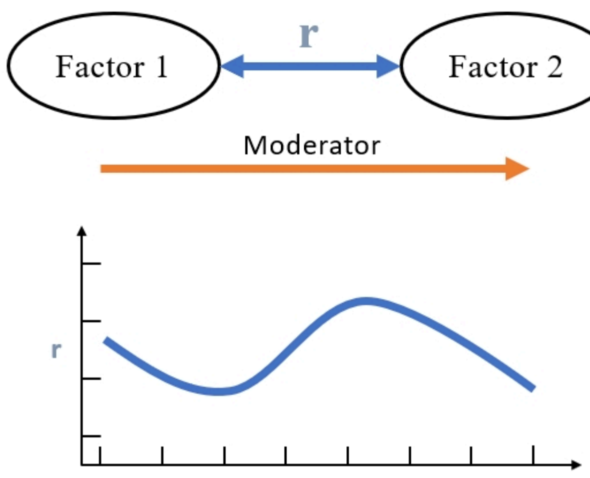
Local SEM

1:00–16:00

Non-Parametric Nonlinear Moderation in SEM

The nonlinear moderation effects in SEM:

- Do parameters of a Structural Equation Model differ across a moderator?



Conceptual Representation

Importantly: No functional assumption about the shape of parameter differences across the moderator (non-parametric)

Nonlinear Moderation Effects in SEM

$$\rho = \rho(M) + \varepsilon$$

- ρ: latent correlation
- M: moderator
- $\rho(M)$: nonlinear function of M

We could use the local model to estimate the $\rho(M)$. Easily speaking, we have a local regression with M as the IV and ρ as DV.

Example 2.3: Parent Depression and Internalizing in Adolescents (pure)

We are studying the **correlation** between latent variables of parent depression (Ds, ξ_1) and adolescents' internalizing problems (Is, ξ_2).

This correlation is influenced nonlinearly by adolescents' age (M).

Local Structural Equation Modeling

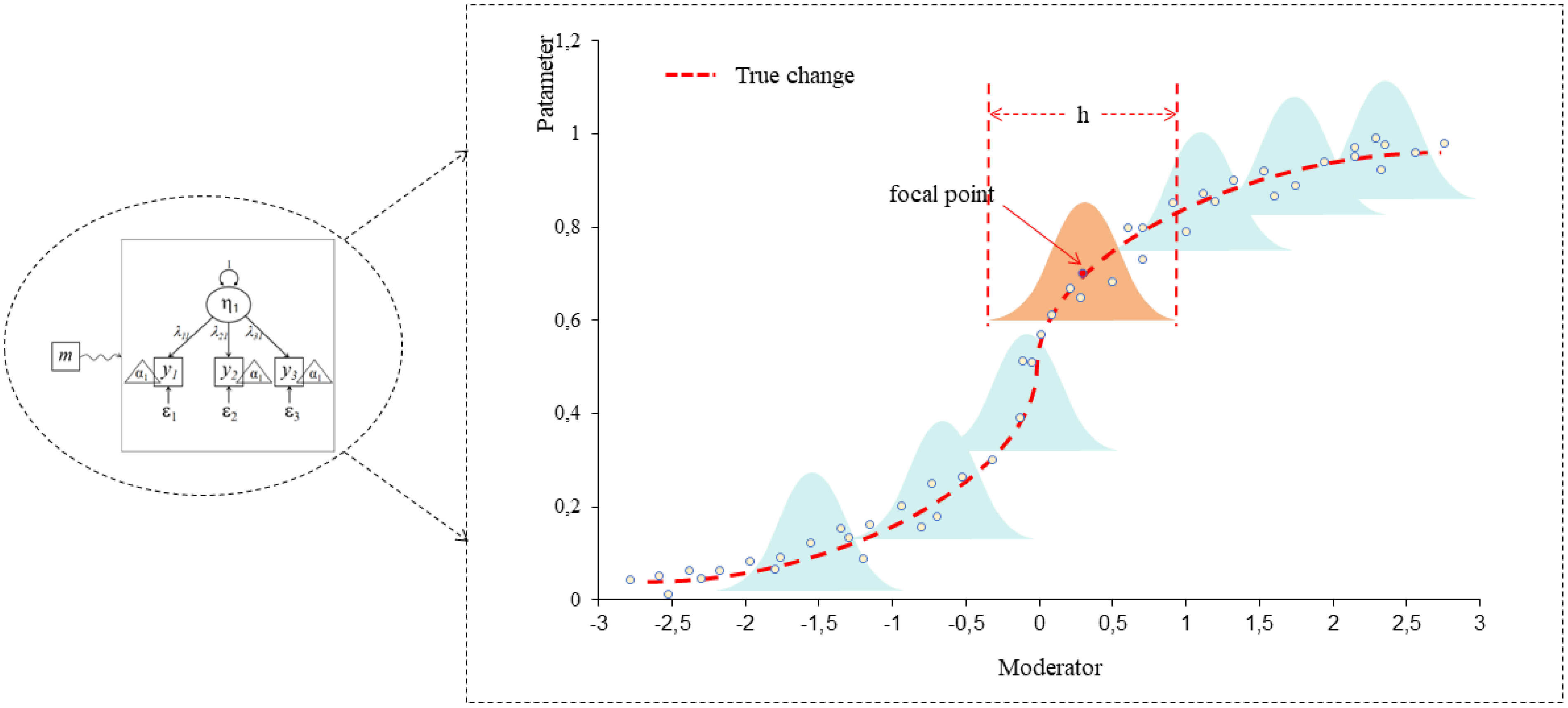
First showed in Hildebrandt et al. (2009), shortly termed as LSEM or LOSEM (Briley et al., 2015)

Same logic as other local models for moderation effects:

- Data-Borrowing and weight the observation at each focal value of the moderator → weighted covariance matrix and mean structure
- Using this information to estimate SEM model in each focal value of the moderator and extract parameters (e.g., ρ)
- Vector of all ρ from all SEM models → non-parametric function of the moderation effect of ρ

Local Structural Equation Modeling

C.
Conceptual diagram for LSEM



LSEM: Separate vs. Joint Estimation

Separate (Pointwise):

- Independently fit model for each focal point of the moderator
- Model fit information is computed at each focal value
- Can not carry out any inference across the moderator (confidence interval, hypothesis tests, etc.)

Joint:

- Single estimation function of all focal values of the moderator
- Global model fit information is computed (same idea as multi-group SEM)
- Could pose constraints among parameters across values of the moderator (many applications possible)
- Automatic if constraints are already in the model using R

LSEM: MI

Measurement invariance is a specific type of moderation that impacts only measurement model parameters (factor loading, intercepts, and residual variances, Hildebrandt et al., 2016)

- We could fit local models with parameter freely estimation and constrained to be equal across moderator values
- Model comparison via different fit information (e.g., CFI, TLI, RMSEA, SRMR)
- For the fit information comparsion recommendation, please refer to Putnick & Bornstein (2016)

For the sake of simplicity, in our workshop, we assume scalar invariance of both latent variables across the different values of adolescent age, which is the minimum level required for comparing latent means and variances (Van de Schoot et al., 2012).

LSEM: Model Specification

- Using a very strong R package `sirt`:

```r
sirt::lsem.estimate(
    data, lavmodel, moderator,
    moderator.grid, h, standardized, meanstructure,
    par_invariant, ...
)

?sirt::lsem.estimate # to get a complete overview of arguments
```

- **Arguments:**
  - `data`, `standardized`, `meanstructure`: same as `lavaan`
  - `lavmodel`: model in `lavaan` syntax
  - `moderator`: name of moderator variable
  - `moderator.grid`: the grid of focal values of the moderator (to save time, we do not need to fit SEM in each value of moderator)
  - h: smoothing index, default = `1.1`
  - `par_invariant`: invariance constraints for parameters

LSEM: lavaan Syntax

```r
load("data/pure.RData")

library(lavaan)
library(sirt)

# Measurement model
model_loadings <- "
  Depression =~ l_dep1 * pdep1 + l_dep2 * pdep2 + l_dep3 * pdep3 + l_dep4 * pdep4 +
                l_dep5 * pdep5 + l_dep6 * pdep6 + l_dep7 * pdep7 + l_dep8 * pdep8 +
                l_dep9 * pdep9

  Internalizing =~ l_int1 * cint1 + l_int2 * cint2 + l_int3 * cint3 +
                   l_int4 * cint4 + l_int5 * cint5
"

model_latent <- "
  Depression ~~ Internalizing
"
```

1 2 3

LSEM: lavaan Syntax

```
model_intercepts <- "
  pdep1 ~ i_dep1 * 1
  pdep2 ~ i_dep2 * 1
  pdep3 ~ i_dep3 * 1
  pdep4 ~ i_dep4 * 1
  pdep5 ~ i_dep5 * 1
  pdep6 ~ i_dep6 * 1
  pdep7 ~ i_dep7 * 1
  pdep8 ~ i_dep8 * 1
  pdep9 ~ i_dep9 * 1

  cint1 ~ i_int1 * 1
  cint2 ~ i_int2 * 1
  cint3 ~ i_int3 * 1
  cint4 ~ i_int4 * 1
  cint5 ~ i_int5 * 1
"
```

LSEM: `lavaan` Syntax

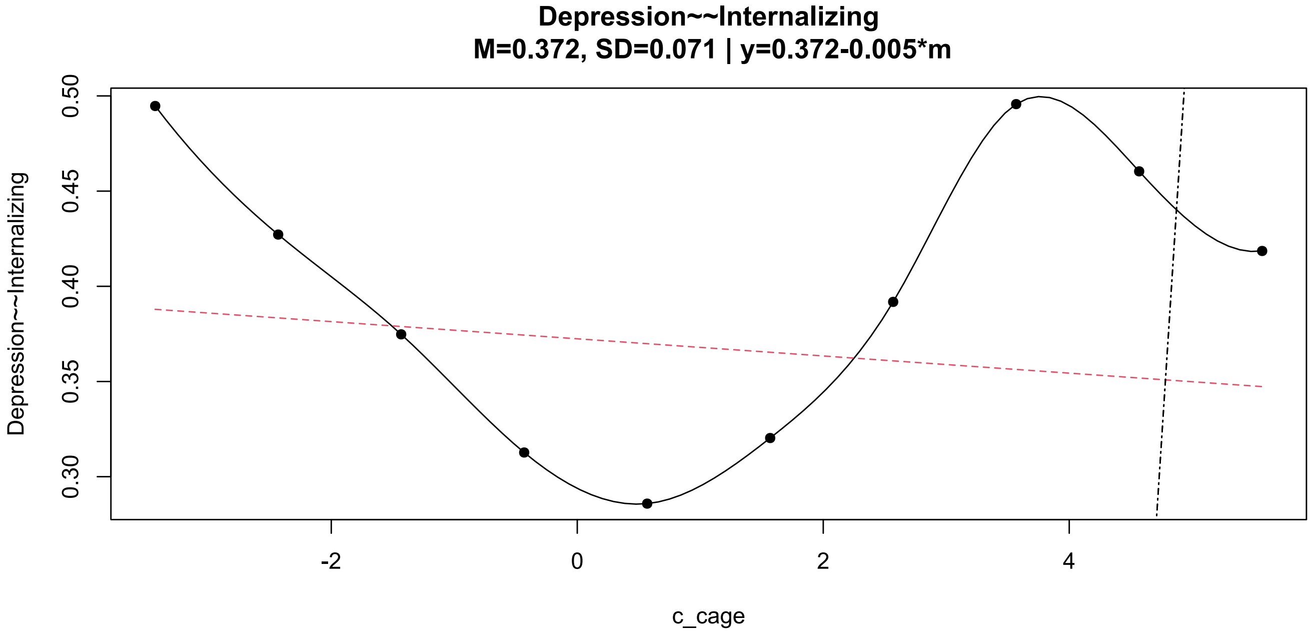
```r
indicators_dep <- paste0("pdep", 1:9)                                                    (1)
indicators_int <- paste0("cint", 1:5)

weak_constraints <- c(paste0("Depression=~", indicators_dep), paste0("Internalizing=~", indicators_int))  (2)
strong_constraints <- c(weak_constraints, paste0(indicators_dep, "~1"), paste0(indicators_int, "~1"))     (3)

model_lsem <- paste0(model_loadings, model_covariances, model_intercepts, collapse = "\n")  (4)

fit_lsem <-                                                                               (5)
  lsem.estimate(
    data = pure,
    lavmodel = model_lsem,
    moderator = "c_cage",
    moderator.grid = seq(min(pure$c_cage), max(pure$c_cage), 1), h = 2,
    standardized = TRUE, std.lv = TRUE, meanstructure = TRUE,                             (6)
    par_invariant = strong_constraints,                                                   (7)
  )

summary(fit_lsem)                                                                         (8)
plot(fit_lsem, parindex = 1)                                                              (9)
```

LSEM: Bootstrapping

You can explain the fit information here as normal SEM:

```
#>          stat      value
#> rmsea rmsea 0.06057355
#> cfi     cfi 0.95556666
#> tli     tli 0.96004512
#> gfi     gfi 0.97931532
#> srmr   srmr 0.05345696
```

You can also see the parameter curve (but without any hypothesis tests):



LSEM: Hypotheses Testing

However, if you want to see the significance of parameter variance across moderator values (test of parameter equivalence), you need compare the following nested bootstrap (same idea as F-test):

- **Null model** (consistent parameter): $\rho = \rho_0 + \varepsilon$
- **Local model**: $\rho = \rho(M) + \varepsilon$
- F-test is difficult to use directly in SEM, because SEM is a multivariate model (more than one outcome variable), so the RSS is difficult to define.
- We could use model comparison with parameter constraints like MI.
- However, we can not get a p-value – even if models are nested, because the joint estimation is a pseudo-likelihood estimation.
  - But we could compare RMEEA, CFI, and SRMR.

LSEM: Hypotheses Testing

However, we could transfer this question to the question about parameter curve distribution only:

- Test of parameter equivalence across M can be viewed as a one-sample t-test of parameter variance in parameter curve distribution (H_0: parameter variance is 0)
- In this way, we only need the estimation of the standard error of the parameter variance

$$t = \frac{\hat{\theta} - \theta_0}{\mathrm{SE}(\hat{\theta})}$$

$\hat{\theta}$: parameter estimate, θ_0: null hypothesis value (0), $\mathrm{SE}(\hat{\theta})$: standard error of the estimate

Nonparametric bootstrap implemented for SE estimation as the naive SE estimate will be biased due to the overlapping (the information of one sample will be used many times due to weighting, Robitzsch, 2023).

LSEM: Bootstrapping

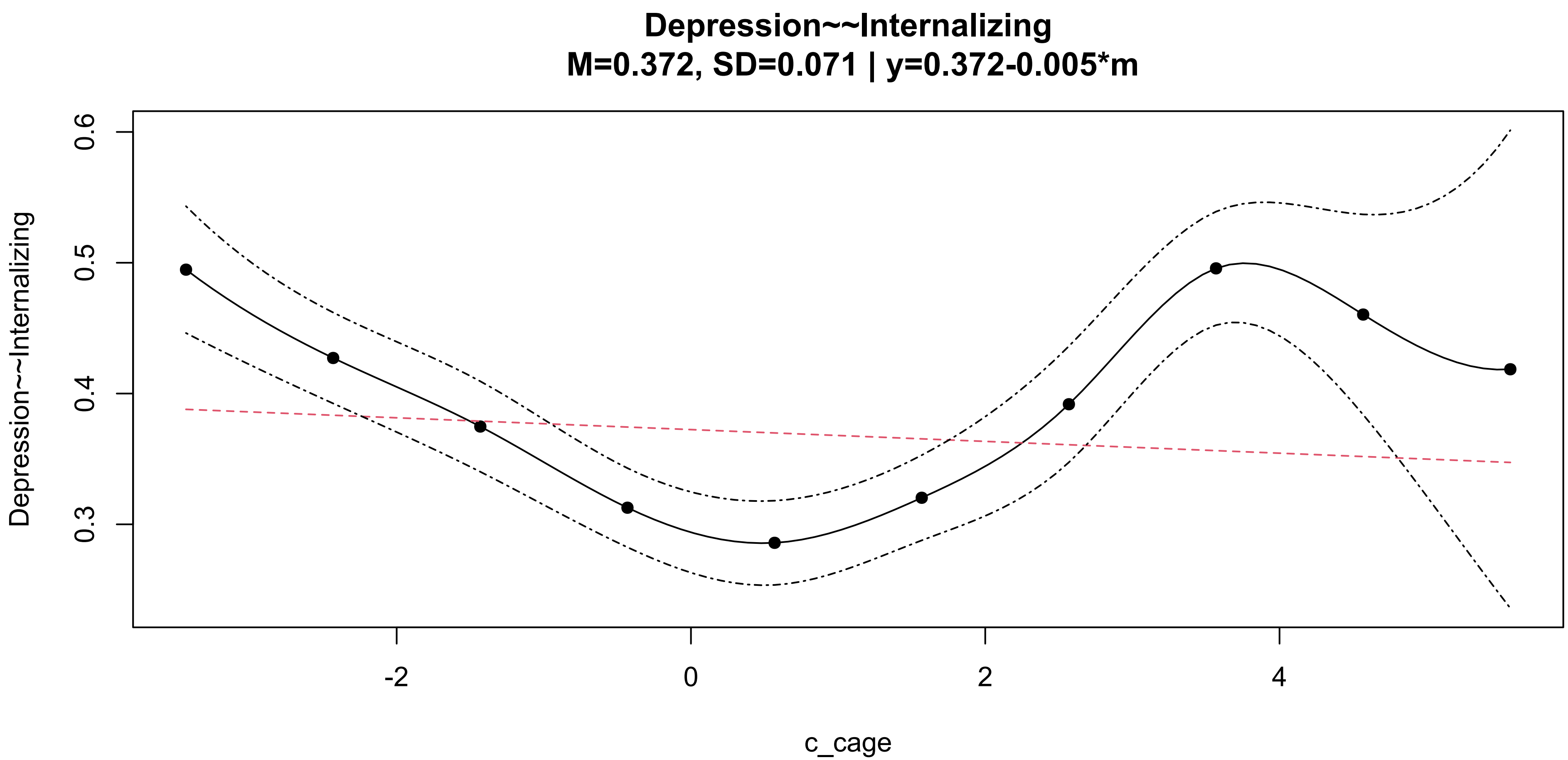
```r
sirt::lsem.bootstrap(object, R, n.core, ...),
?sirt::lsem.bootstrap # to get a complete argument
```

- **Arguments:**
  - `object`: object from `lsem.estimate()`
  - R: expected number of bootstrap samples
  - `n.core`: number of cores for parallel computing

```r
boot_lsem <- lsem.bootstrap(object = fit_lsem, R = 100, n.core = 4)
summary(boot_lsem)
plot(boot_lsem, parindex = 1)
```

LSEM: Bootstrapping

```
#>                          par parindex      M      SD   SD_bc   SD_se  SD_t
#> 1 Depression~~Internalizing        1 0.3724 0.07117 0.05987 0.00749 7.993
#>        SD_p     MAD    Min    Max lin_int   lin_slo SD_nonlin
#> 1 6.561e-16 0.05966 0.2859 0.4957  0.3724 -0.004509   0.07052
```



LSEM: Parametric Moderation

Similar to any other local models, we could test if this local non-linear model fits the moderation effects better than a parametric (linear, quadratic) moderation model (MNSEM):

$$\rho = \rho(M) + \varepsilon$$

- **Linear Model:** $\rho(M) = \rho_0 + \rho_1 M$
- **Quadratic Model:** $\rho(M) = \rho_0 + \rho_1 M + \rho_1 M^2$
- **Local Model:** $\rho(M)$ is a nonlinear, nonparametric function of M

Based on the same idea as the F-test in local regression, we could test the parametric function based on the parameter curve distribution.

LSEM: Parametric Moderation

```r
sirt::lsem.test(mod, bmod, models, ...)
```

- **Arguments:**
  - mod: object from `lsem.estimate()`
  - bmod: object from `lsem.bootstrap()`
  - `models`: the aimed parametric moderation

Because the correlation is a bounded parameter, i.e., [-1, +1], we need Fisher's z-transformation to bound the parameter:

```r
test_linear <- list("Depression~~Internalizing" = 0.5 * log((1 + y) / (1 - y)) ~ m)
test_quadratic <- list("Depression~~Internalizing" = 0.5 * log((1 + y) / (1 - y)) ~ m + I(m^2))

tmod_linear <- lsem.test(mod = fit_lsem, bmod = boot_lsem, models = test_linear)
tmod_quadratic <- lsem.test(mod = fit_lsem, bmod = boot_lsem, models = test_quadratic)
```

LSEM: Parametric Moderation

```
print(tmod_linear$test_models, digits = 4)
```

```
#>                          par        coef       est        se       t           p
#> 1 Depression~~Internalizing (Intercept)  0.393971 0.010663 36.9479 7.883e-299
#> 2 Depression~~Internalizing           m -0.005296 0.007662 -0.6912  4.895e-01
#>   chisq_het df_het  p_het chisq_fit df_fit     p_fit
#> 1    0.4777      1 0.4895     91.06      8 3.331e-15
#> 2        NA     NA     NA        NA     NA        NA
```

- t-test for each coefficient
- `chisq_het` (test for parameter heterogeneity of parametric moderation): tests for multiple parameters (here, only linear) different from 0 simultaneously
- `chisq_fit` (test of a sufficient fit of a parameter curve): tests if the model fit of the parametric model is significant from the the local model (underfitting)

LSEM: Parametric Moderation

```
print(tmod_quadratic$test_models, digits = 4)
```

```
#>                             par        coef      est        se       t          p
#> 1 Depression~~Internalizing (Intercept)  0.32549 0.019900 16.356 3.920e-60
#> 2 Depression~~Internalizing           m -0.01362 0.005235 -2.601 9.304e-03
#> 3 Depression~~Internalizing      I(m^2)  0.01501 0.003075  4.879 1.064e-06
#>   chisq_het df_het     p_het chisq_fit df_fit   p_fit
#> 1     26.99      2 1.376e-06     20.05      7 0.02875
#> 2        NA     NA        NA        NA     NA      NA
#> 3        NA     NA        NA        NA     NA      NA
```

- t-test for each coefficient
- `chisq_het` (test for parameter heterogeneity of parametric moderation): tests for multiple parameters (here, linear and quadratic) different from 0 simultaneously
- `chisq_fit` (test of a sufficient fit of a parameter curve): tests if the model fit of the parametric model is significant from the the local model (n.s., thus quadratic fit is sufficient)

Session 5

Advanced Topics and Recommendations

17:00–17:45

Advanced Topics

For an overview, read the paper for the following topics:

- Effect coding
- Categorical variables
- Missing values
- Statistical power
- Multiple moderators or latent moderators
- Advanced SEMs
- Alternative method for bootstrapping
- Regularization

Liu, T., Ding, R., Su, Z., Peng, Z., & Hildebrandt, A. (accepted). Modeling nonlinear moderation effects with local structural equation modeling (LSEM): A non-technical introduction. *International Journal of Psychology*.

Measurement Invariance

> **Fixed Parameters and Measurement Invariance**
>
> With ULI and UVI, fixed parameters are already assumed to be measurement invariant (Kline, 2023), i.e., the fixed parameters (loading of reference indication or latent variance) are assumed to be measurement invariant.

One way to solve this problem is effect coding identification (Little et al., 2006): It constrains the indicators' …

- **loadings** to sum up to the **number** of **indicators** for each latent variable
- **intercepts** to sum up to **0** for each latent factor.

Under effect coding, no arbitrary decisions about which parameter should be fixed, i.e., no arbitrary assumptions on measurement invariance

> **Bifactor-Like Models and Models with Cross-Loading**
>
> Do not use effect coding if you have a bifactor-like model or model with cross-loading (Graves & Merkle, 2022)!

Example: Effect Coding

```
model_loadings <- "
  # lavaan used ULI as default for factor loading
  # We force lavaan to freely estimate the loading of the first indicator
  # We give each loading a name for constraints
  Depression =~ NA * pdep1 + l_dep1 * pdep1 + l_dep2 * pdep2 + l_dep3 * pdep3 + l_dep4 * pdep4 + l_dep5 * pde
                l_dep6 * pdep6 + l_dep7 * pdep7 + l_dep8 * pdep8 + l_dep9 * pdep9
  Internalizing =~ NA * cint1 + l_int1 * cint1 + l_int2 * cint2 + l_int3 * cint3 + l_int4 * cint4 + l_int5 *

  # Constraint: Sum of loadings
  l_dep1 == 9 - l_dep2 - l_dep3 - l_dep4 - l_dep5 - l_dep6 - l_dep7 - l_dep8 - l_dep9
  l_int1 == 5 - l_int2 - l_int2 - l_int3 - l_int4
"
```

Example: Effect Coding

```
model_intercepts <- "
  # We give each intercept a name for constraints
  pdep1 ~ i_dep1 * 1
  pdep2 ~ i_dep2 * 1
  pdep3 ~ i_dep3 * 1
  pdep4 ~ i_dep4 * 1
  pdep5 ~ i_dep5 * 1
  pdep6 ~ i_dep6 * 1
  pdep7 ~ i_dep7 * 1
  pdep8 ~ i_dep8 * 1
  pdep9 ~ i_dep9 * 1
  cint1 ~ i_int1 * 1
  cint2 ~ i_int2 * 1
  cint3 ~ i_int3 * 1
  cint4 ~ i_int4 * 1
  cint5 ~ i_int5 * 1

# Constraint: Sum of intercepts
  l_dep1 == 0 - l_dep2 - l_dep3 - l_dep4 - l_dep5 - l_dep6 - l_dep7 - l_dep8 - l_dep9
  l_int1 == 0 - l_int2 - l_int3 - l_int4 - l_int5
"
```

Example: Effect Coding

```r
model_latent <- "
  # lavaan used UVI as default for intercept
  # We force lavaan to estimate the factor means
  Depression ~ NA * 1
  Internalizing ~ NA * 1

  # Latent Covariance
  Depression ~~ Internalizing
"

fit <- sem(model = paste0(model_loadings, model_intercepts, model_latent), data = pure)
```

Latent correlation is unaltered:

```r
summary(fit, standardized = TRUE, fit.measures = TRUE)
```

```
#> Covariances:
#>                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
#>   Depression ~~
#>     Internalizing     0.000    0.000    2.543    0.011    0.010    0.010
```

Example: Effect Coding

Fit indices are not altered (trimmed output):

```
#> User Model versus Baseline Model:
#>
#>   Comparative Fit Index (CFI)                    0.751
#>   Tucker-Lewis Index (TLI)                       0.698
#>
#> Loglikelihood and Information Criteria:
#>
#>   Akaike (AIC)                              409617.578
#>   Bayesian (BIC)                            409964.361
#>   Sample-size adjusted Bayesian (SABIC)     409824.531
#>
#> Root Mean Square Error of Approximation:
#>
#>   RMSEA                                          0.166
#>   90 Percent confidence interval - lower         0.165
#>   90 Percent confidence interval - upper         0.168
#>   P-value H_0: RMSEA <= 0.050                    0.000
#>   P-value H_0: RMSEA >= 0.080                    1.000
#>
#> Standardized Root Mean Square Residual:
#>
#>   SRMR                                           0.204
```

Categorical Variables

- Categorical **moderator**: Multiple-Group SEM
- Categorical **observed** exogenous **variable**: contrast coding (e.g., dummy coding)
- Categorical exogenous **indicator** in SEM:
  - **MNSEM:** weighted least squares mean and variance adjusted (WLSMV) or maximum likelihood with robust standard errors (MLR) estimators is easily implemented in `OpenMx` (Kolbe et al., 2022) or `Mplus` (Chen & Bauer, 2024)
  - **LSEM:** WLSWV and MLR are also easy to implement in `sirt` Panayiotou et al. (2023)

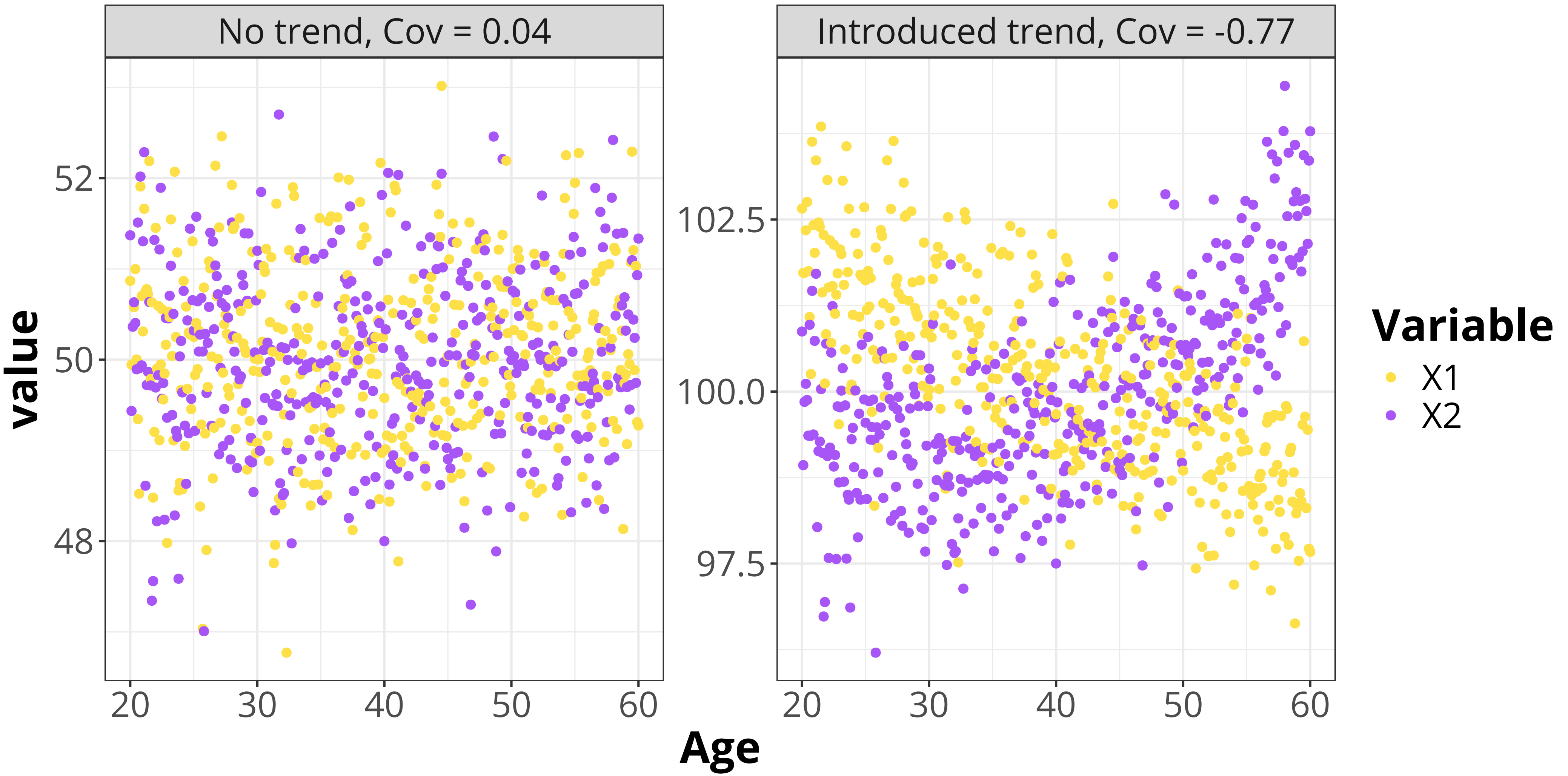
Caution: Categorical Variables

No matter if WLSWV or MLR is used, bias is always present in LSEM estimation. The reason is the mean-induced association confound:

$$\mathrm{Cov}(X, Y) = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

- Two variables may show covariance due to their individual means being influenced by a third variable (e.g., age)
- This covariation reflects changes in the means, not an actual relationship between the variables
- To accurately assess the true relationship of covariance, the trends in the means caused by the third variable must be removed
- However, de-trending is easy to implement in continuous indicator (in `sirt`, `residualize = TRUE` is automatically) instead of categorical indicator

Caution: Categorical Variables



Data without Trend vs. Data with Introduced Trend

Missing Values

Missing Completely At Random (MCAR) in SEM:

- Pairwise deletion is usually used
- Full-information ML (FIML) can be used for efficiency reasons (smaller variance of estimates)

Missing At Random (MAR) in SEM:

- **FIML** is usually recommended for MAR data (Enders & Bandalos, 2001)
  - **MNSEM**: readily implemented in `OpenMx` (Kolbe et al., 2022) or `Mplus` (Chen & Bauer, 2024)
  - **LSEM**: readily implemented in `sirt`
    - Note: FIML can bias LSEM estimation if both mean structure and covariance matrices are applied as trends in the mean structure may distort covariance matrix estimation (Robitzsch, 2023)

Missing Values

Missing At Random (MAR) in SEM:

- **Multiple imputation** as an alternative for FIML
  - **MNSEM:** using Rubin's rules to integrate coefficient from different imputation results (Feldon et al., 2024)
  - **LSEM:** integrating multiple imputation results using Rubin's rules remains complex in LSEM (Basarkod et al., 2023)

Decision Tree in LSEM based on the importance of **mean structure estimation:**

- Mean structure **not important**: FIML with the `meanstructure = FALSE` in LSEM
- Mean structure **important** (e.g., strong measurement invariance): multiple imputations with "manually combined" strategy in LSEM

Missing value in **moderator:**

- **Multiple imputation** only: under development

Statistical Power

For **LSEM**:

- the effective N of the smallest focal point should be big enough

For **MNLFA**:

- needs less sample size than LSEM

> **Sample Size Criteria**
>
> To our knowing, there are no standard criteria for sample size. To check if your sample size is sufficient to detect an effect of a moderation of a specific size, a Monte Carlo simulation could be run.

Statistical Power: LSEM

Distribution of moderator (density and ESS):

```
1  fit_lsem$moderator.density
```

```
#>     moderator        wgt      Neff
#> 1  -3.4315068 0.09844177 2363.0394
#> 2  -2.4315068 0.09074876 3063.1058
#> 3  -1.4315068 0.17359061 4700.3577
#> 4  -0.4315068 0.18333846 5114.4434
#> 5   0.5684932 0.14840554 4326.7897
#> 6   1.5684932 0.11575760 3415.6613
#> 7   2.5684932 0.09306339 2675.9010
#> 8   3.5684932 0.06374307 1831.2803
#> 9   4.5684932 0.02966693  900.6200
#> 10  5.5684932 0.00324386  206.4214
```

```
1  fit_lsem$moderator.stat
```

```
#>    variable             M           SD          min          max
#> 1 moderator -1.924698e-16 2.135965e+00  -3.43150685    5.5684932
#> 2       wgt  1.000000e-01 5.841562e-02   0.00324386    0.1833385
#> 3      Neff  2.859762e+03 1.605224e+03 206.42144797 5114.4433743
```

Effective Degrees of Freedom

Effective degrees of freedom df measures the overall complexity of the local model, reflecting how flexible the model is in fitting the data.

- **High** EDF: Indicates that the model is very flexible at the given focal value.
- **Low** EDF: Suggests that the model is smoother at the given focal value.

For each data point x_i, the local EDF is the trace of the local hat matrix $\mathbf{H}_i$, the total EDF for the entire model is the sum of all local EDFs or the trace of the global hat matrix $\mathbf{H}$:

$$\mathrm{EDF}_{\mathrm{total}} = \sum_{i=1}^{n} \mathrm{EDF}_i = \mathrm{trace}(\mathbf{H})$$

EDF is closely related to the model's bias-variance trade-off and model comparsion.

- **High** EDF may lead to overfitting, capturing noise in the data.
- **Low** EDF may lead to underfitting, missing important patterns in the data.

Multiple Moderators

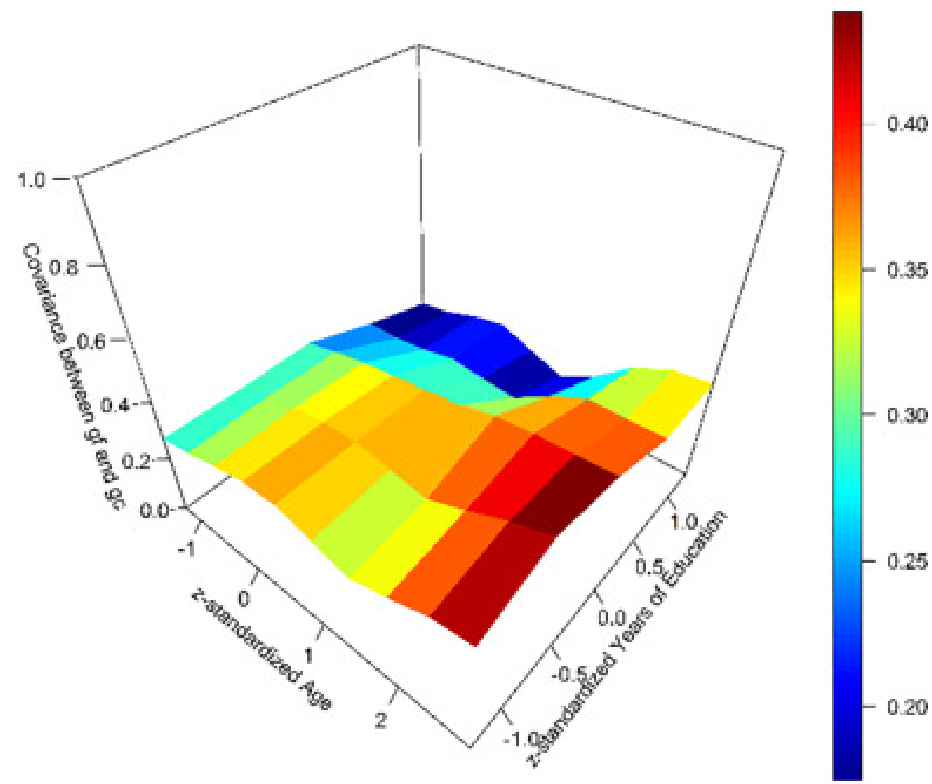
MNSEM:

- **Straightforward** addition of more covariates to **defined variables**
  - Main (polynomial) effects of other covariates
  - Interaction effects between covariates
- **Implementation example:** Kolbe et al. (2022)
  - Effects of **age** and **gender** on a latent correlation (with measurement invariance tests)
  - Also part of the `mxsem` tutorial
- **Example:**
  - **Equation:** $\beta_1(M_1, M_2) = \beta_{10} + \beta_{11} M_1 + \beta_{12} + M_2$
  - **Code:** `b1 := b10 + b11 * data.age + b12 * data.gender`

Multiple Moderators

LSEM: Multivariate Local Structural Equation Model (Hartung et al., 2018)

- Two moderators using multivariate kernel
- Parameter changes across both moderators, including their interaction



Regression Plane of a Latent Covariance for 2 Moderators (Hartung et al., 2018)

Latent Moderators

For **MNSEM**: theoretically possible to implement using MPlus

- Caveat: estimation demand would be a big issue

For **LSEM**: recommended 2-step approach for including the latent moderator:

1. Estimate latent **factor scores** by item response theory (IRT) / factor models (for **reflective** measurement model), or **component scores** by principal component analysis (for **formative measurement model**)
2. Use **estimated score** as manifest moderator
   - For **reflective** measurement models: expected a posteriori (EAP) factor score could return an unbiased estimation in LSEM

Alternative Method for Bootstrapping

In **LSEM**, besides bootstrapping, one could also use permutation testing (Hildebrandt et al., 2016):

- Number of “permutation” data sets in which moderator values are randomly assigned to participants → model estimates independent of the moderator variable
- Test if parameters deviate from parameter distribution across permutations

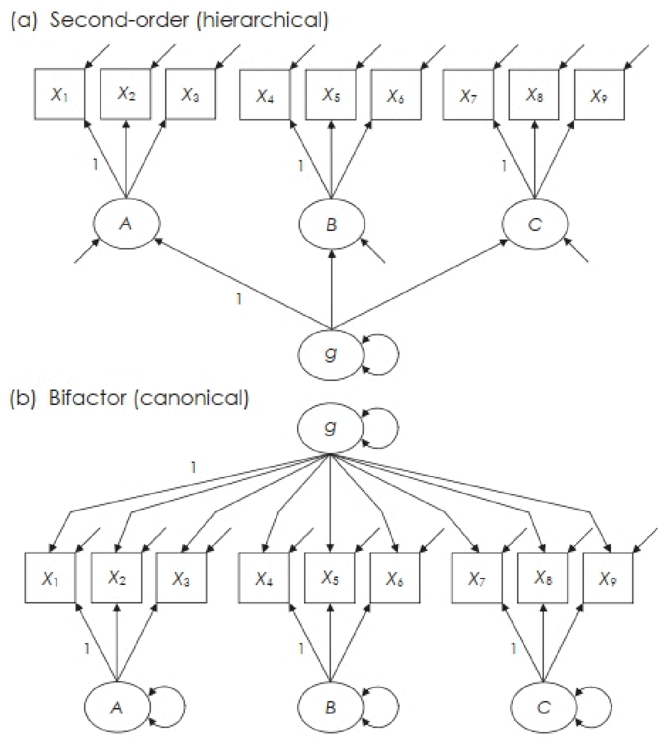
Comparing **bootstrapping** and **permutation** testing:

- **Bootstrapping:** depend on t-statistics, which assumes a normal distribution of parameter vector across moderator values (Robitzsch, 2023)
- **Permutation:** distribution-free statistics, test all parameters simultaneously by constraint all parameter invariant
  - Difficult for confirmatory testing of a hypothesized parametric moderation effect

Advanced SEMs

Bi-factor/higher-order model:

- **MNSEM**: e.g., Morgan-López et al. (2024)
- **LSEM**: e.g., Schroeders & Jansen (2022)

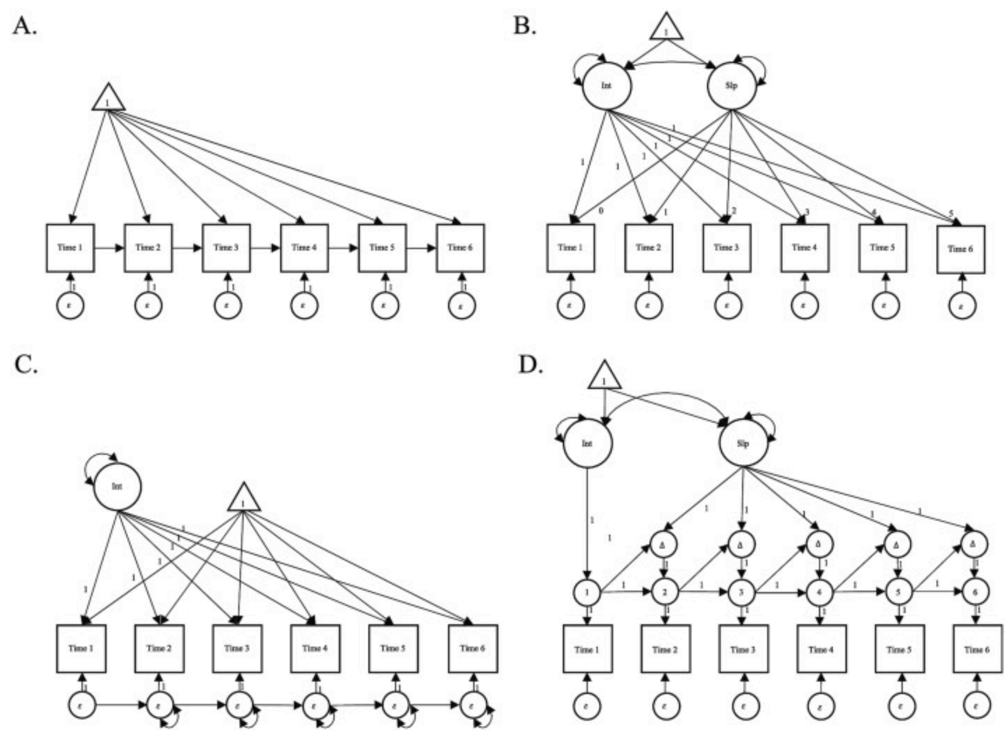


Higher-order models as shown in Kline (2023)

Advanced SEMs

Longitudinal models:

- **MNSEM**: e.g., DeJoseph et al. (2021)
- **LSEM**: e.g., Olaru et al. (2023)



A = Autoregressive Panel Model, B = Linear Latent Growth Curve Model, C = Random-Intercepts Panel Model, D = Dual-Change Latent Change Score Model. Int = intercept, Slp = slope. Unlabeled paths are estimated.

Reminder: Model Complexity

- **Trade-off:**
  - **Local models:** tend become **too complex** models → potential overfit
  - **Polynomial models:** tend to be **too simple** models → potential underfit
    - MNSEM models could also very easily become too complex
- **Model simplification:** approaches to avoid overfit
  - **Check significance:** non-significant (especially higher-order) terms could be dropped
  - **Regularization techniques** (e.g., LASSO or ridge regression) to shrink coefficients of less important variables to 0
    - For an overview on regularization in MNLFA, see Bauer et al. (2020)
  - **Cross-validation:** ensure simpler model is robust and does not overfit

Regularization

- Simple example from regression: model trimming via regularization:

```r
library(glmnet)
set.seed(123)
X <- model.matrix(
    aca_per ~ 1 + sle_qua_c + I(sle_qua_c^2) +
        stu_hou_c + sle_qua_c:stu_hou_c + I(sle_qua_c^2):stu_hou_c,
    data = performance
)[, -1]
y <- performance$aca_per
fit0 <- glmnet(X, y, alpha = 0.5)
cv_fit <- cv.glmnet(X, y, alpha = 0.5)
fit <- glmnet(X, y, alpha = 0.5, lambda = cv_fit$lambda.1se)
coef(fit)
```

```
#> 6 x 1 sparse Matrix of class "dgCMatrix"
#>                                   s0
#> (Intercept)               41.9279913
#> sle_qua_c                  0.2679986
#> I(sle_qua_c^2)             .
#> stu_hou_c                  .
#> sle_qua_c:stu_hou_c        0.2554464
#> I(sle_qua_c^2):stu_hou_c -0.1001003
```

Also applicable to OpenMx (see this tutorial) and could potentially be implemented with `mxsem`

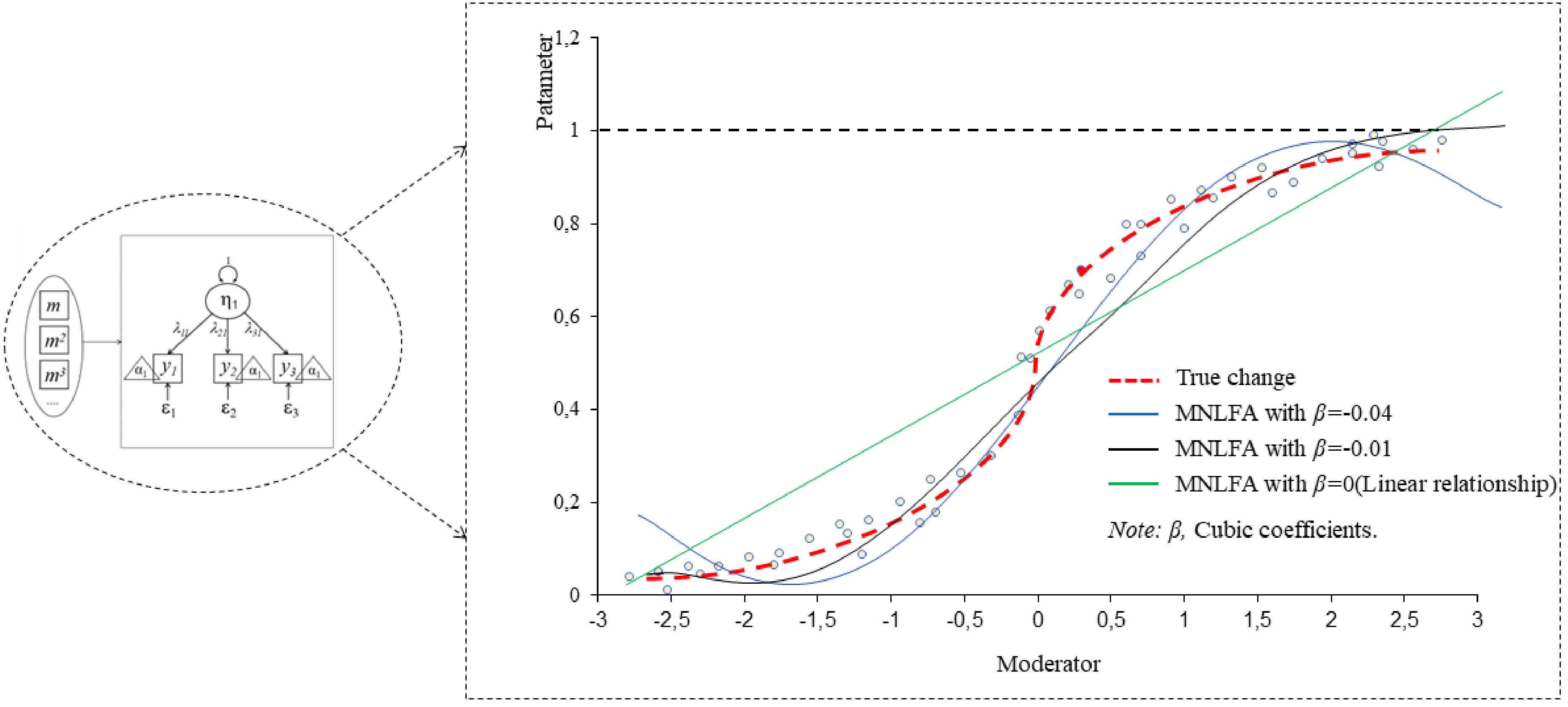
Comparing Global vs. Local Model

- Model nonlinear moderation effects SEMs in with **LSEM or MNSEM**?
  - **Simulation study** (on structural coefficients):
    - **Parameters:**
      - Moderator effect **shapes:** linear vs. quadratic vs. logistic
      - Moderator effect **sizes:** small vs. medium
      - **Sample sizes:** 500 vs. 1 000 vs. 2 000

Moderated Nonlinear Factor Analysis

B.

Conceptual diagram for MNLFA(take the cubic relationship as an example)



Conceptual Representation

Comparing Global vs. Local Model

- **Simulation study** (on structural coefficients):
  - **Results:**
    - Effects of **effect shape:**
      - **Bias:**
        - Correct shape **mirrored** in MNSEM: **MNSEM** > LSEM
        - Correct (esp. logistic) shape **not mirrored** in MNSEM: **LSEM** > MNSEM
      - **RMSE (efficiency):**
        - **MNSEM** is **more efficient** estimators (higher RMSE) than LSEM
    - Effects of **sample size** and **effect size:**
      - Both models perform **better** under **larger sample sizes** and under **weaker moderator effects**
- **Careful extrapolation:** the results should generalize to regression models

Model Selection

- **Recommendation:**
  - **Known shape** of moderator effect: **parametric** (global) model, i.e., **moderated nonlinear** regression and **MNSEM**
  - **Unknown shape** of moderator effect: **local** model, i.e., **local** regression and **LSEM**
- **Source of information** on shape of the moderation effect:
  - **Confirmatory** approach: clear idea of expected shape (previous literature) → MNSEM (because of lower sample size requirements) or LSEM (if sample size is not a problem; confirmatory hypothesis tests available)
  - **Exploratory** approach: LSEM, followed by parametric tests and plots → foundation for moderation effects

Model Selection

- One last note about using **previous literature** as a source of information for MNSEM:
  - If MNSEM was used, the used shape might already be biased
    - **Example:** authors used a **cubic** shape, whereas the true relation might be **logistic**
  - Bias might be small, but this might cause theoretical misinterpretations
    - **Example: logistic** curves are **bounded**, whereas **cubic** curves are unbounded

Questions and Feedback

17:45–18:00

Your Feedback

https://survey.studiumdigitale.uni-frankfurt.de/nonlinear_moderation/

References

Andersen, R. (2009). Nonparametric methods for modeling nonlinearity in regression analysis. *Annual Review of Sociology*, *35*(1), 67–85.

Awang, Z., Wan Afthanorhan, W. M. A., & Asri, M. A. M. (2015). Parametric and non parametric approach in structural equation modeling (SEM): The application of bootstrapping. *Mod. Appl. Sci.*, *9*(9).

Basarkod, G., Marsh, H. W., Sahdra, B. K., Parker, P. D., Guo, J., Dicke, T., & Lüdtke, O. (2023). The dimensionality of reading self-concept: Examining its stability using local structural equation models. *Assessment*, *30*(3), 873–890.

Bauer, D. J. (2017). A more general model for testing measurement invariance and differential item functioning. *Psychological Methods*, *22*(3), 507.

Bauer, D. J., Belzak, W. C. M., & Cole, V. T. (2020). Simplifying the assessment of measurement invariance over multiple background variables: Using regularized moderated nonlinear factor analysis to detect differential item functioning. *Struct. Equ. Modeling*, *27*(1), 43–55.

Bauer, D. J., & Hussong, A. M. (2009). Psychometric approaches for developing commensurate measures across independent studies: Traditional and new

models. *Psychol. Methods*, *14*(2), 101–125.
Boker, S., Neale, M., Maes, H., Wilde, M., Spiegel, M., Brick, T., Spies, J., Estabrook, R., Kenny, S., Bates, T., Mehta, P., & Fox, J. (2011). OpenMx: An open source extended structural equation modeling framework. *Psychometrika*, *76*(2), 306–317.
Borsboom, D. (2006). When does measurement invariance matter? *Medical Care*, *44*(11), S176–S181.
Briley, D. A., Harden, K. P., Bates, T. C., & Tucker-Drob, E. M. (2015). Nonparametric estimates of gene$\times$ environment interaction using local structural equation modeling. *Behavior Genetics*, *45*, 581–596.
Byrne, B. M. (1994). *Structural equation modeling with EQS and EQS/windows*. Sage.
Chen, S. M., & Bauer, D. J. (2024). Modeling construct change over time amidst potential changes in construct measurement: A longitudinal moderated factor analysis approach. *Psychological Methods*.
Curran, P. J., McGinley, J. S., Bauer, D. J., Hussong, A. M., Burns, A., Chassin, L., Sher, K., & Zucker, R. (2014). A moderated nonlinear factor model for the development of commensurate measures in integrative data analysis. *Multivariate Behavior Research*, *49*(3), 214–231.
DeJoseph, M. L., Sifre, R. D., Raver, C. C., Blair, C. B., & Berry, D. (2021). Capturing environmental dimensions of adversity and resources in the context of poverty

across infancy through early adolescence: A moderated nonlinear factor model. *Child Development*, *92*(4), e457–e475.

Enders, C. K., & Bandalos, D. L. (2001). The relative performance of full information maximum likelihood estimation for missing data in structural equation models. *Structural Equation Modeling*, *8*(3), 430–457.

Fan, X., Thompson, B., & Wang, L. (1999). Effects of sample size, estimation methods, and model specification on structural equation modeling fit indexes. *Struct. Equ. Modeling*, *6*(1), 56–83.

Feldon, D. F., Kim, Y. M., Litson, K., Ramadan, N., Fields, D., & Brockbank, R. (2024). Idiographic moderation in measurement models. In *Mixed methods for psychological measurement* (pp. 101–126). Routledge.

Gana, K., & Broc, G. (2019). *Structural equation modeling with lavaan*. ISTE Ltd; John Wiley & Sons.

Gottfredson, N. C., Cole, V. T., Giordano, M. L., Bauer, D. J., Hussong, A. M., & Ennett, S. T. (2019). Simplifying the implementation of modern scale scoring methods with an automated R package: Automated moderated nonlinear factor analysis (aMNLFA). *Addict. Behav.*, *94*, 65–73.

Graves, B., & Merkle, E. C. (2022). A note on identification constraints and information criteria in bayesian latent variable models. *Behavior Research Methods*, 1–10.

Guenole, N., & Brown, A. (2014). The consequences of ignoring measurement invariance for path coefficients in structural equation models. *Frontiers in Psychology*, *5*, 980.

Hartung, J., Doebler, P., Schroeders, U., & Wilhelm, O. (2018). Dedifferentiation and differentiation of intelligence in adults across age and years of education. *Intelligence*, *69*, 37–49.

Hastie, T., Tibshirani, R., & Friedman, J. (2017). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Hertzog, C. (2011). Intelligence in adulthood. *The Cambridge Handbook of Intelligence*, 174–190.

Hildebrandt, A., Lüdtke, O., Robitzsch, A., Sommer, C., & Wilhelm, O. (2016). Exploring factor model parameters across continuous variables with local structural equation models. *Multivariate Behavioral Research*, *51*(2-3), 257–258.

Hildebrandt, A., Wilhelm, O., & Robitzsch, A. (2009). Complementary and competing factor analytic approaches for the investigation of measurement invariance. *Review of Psychology*, *16*(2), 87–102.

Hu, L.-T., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Struct. Equ. Modeling*, *6*(1), 1–55.

Jacoby, W. G. (2000). Loess:: A nonparametric, graphical tool for depicting relationships between variables. *Electoral Studies*, *19*(4), 577–613.

Kaltwasser, L., Hildebrandt, A., Recio, G., Wilhelm, O., & Sommer, W. (2014). Neurocognitive mechanisms of individual differences in face cognition: A replication and extension. *Cognitive, Affective, & Behavioral Neuroscience*, *14*, 861–878.

Kline, R. B. (2023). *Principles and practice of structural equation modeling*. Guilford publications.

Kolbe, L., Molenaar, D., Jak, S., & Jorgensen, T. D. (2022). Assessing measurement invariance with moderated nonlinear factor analysis using the r package OpenMx. *Psychological Methods*.

Leung, Y., Mei, C.-L., & Zhang, W.-X. (2000). Statistical tests for spatial nonstationarity based on the geographically weighted regression model. *Environment and Planning A*, *32*(1), 9–32.

Li, Z., & Fotheringham, A. S. (2020). Computational improvements to multi-scale geographically weighted regression. *International Journal of Geographical Information Science*, *34*(7), 1378–1397.

Little, T. D., Slegers, D. W., & Card, N. A. (2006). A non-arbitrary method of identifying and scaling latent variables in SEM and MACS models. *Structural Equation Modeling*, *13*(1), 59–72.

Maassen, E., D'Urso, E. D., Van Assen, M. A., Nuijten, M. B., De Roover, K., & Wicherts, J. M. (2023). The dire disregard of measurement invariance testing in psychological science. *Psychological Methods*.

Molenaar, D., Rózsa, S., & Bolsinova, M. (2019). A heteroscedastic hidden markov mixture model for responses and categorized response times. *Behavior Research Methods*, *51*, 676–696.

Morgan-López, A. A., Saavedra, L. M., McDaniel, H. L., West, S. G., Ialongo, N. S., Bradshaw, C. P., Tonigan, A. T., Montgomery, B. W., Powell, N. P., Qu, L., et al. (2024). Beyond jacobson and truax: Estimation of clinical significance trajectories in the coping power intervention using measurement error-corrected multilevel modeling. *Behavior Therapy*.

Olaru, G., Robitzsch, A., Hildebrandt, A., & Schroeders, U. (2023). An illustration of local structural equation modeling for longitudinal data: Examining differences in competence development in secondary schools. *Education, Competence Development and Career Trajectories*, 153.

Orzek, J. H. (2024). *Mxsem: Specify 'OpenMx' models with a 'lavaan'-style syntax*. https://jhorzek.github.io/mxsem/

Panayiotou, M., Badcock, J. C., Lim, M. H., Banissy, M. J., & Qualter, P. (2023). Measuring loneliness in different age groups: The measurement invariance of the UCLA loneliness scale. *Assessment*, *30*(5), 1688–1715.

Pokropek, A., Davidov, E., & Schmidt, P. (2019). A monte carlo simulation study to assess the appropriateness of traditional and newer approaches to test for measurement invariance. *Structural Equation Modeling: A Multidisciplinary Journal*, *26*(5), 724–744.

Putnick, D. L., & Bornstein, M. H. (2016). Measurement invariance conventions and reporting: The state of the art and future directions for psychological research. *Developmental Review*, *41*, 71–90.

Raj, R. (2024). *Image from "kernel density estimation (KDE): Unraveling the intriguing patterns with 5 numerical experiments"*. https://levelup.gitconnected.com/kernel-density-estimation-kde-unraveling-the-intriguing-patterns-with-5-numerical-experiments-3d4b292a8f68

Rehabilitation Nursing Foundation Courses. (2024). *Image from lesson 7: Summary on rehabilitation nursing foundation courses website.* https://www.rnfc.org/courses/isl/Lesson%207/Summary/

Robitzsch, A. (2023). Estimating local structural equation models. *Journal of Intelligence*, *11*(9), 175.

Schroeders, U., & Jansen, M. (2022). Science self-concept–more than the sum of its parts? *The Journal of Experimental Education*, *90*(2), 435–451.

Schroeders, U., & Wilhelm, O. (2011). Equivalence of reading and listening comprehension across test media. *Educational and Psychological Measurement*, *71*(5), 849–869.