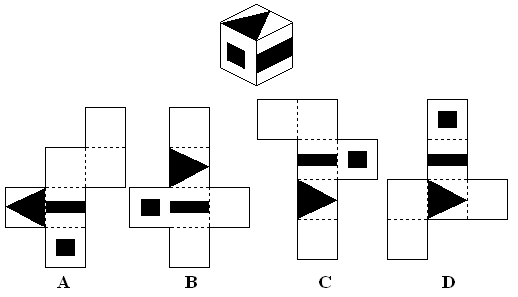
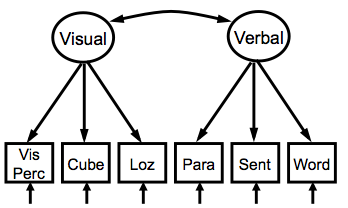
**Reasons for factor analysis**

* To find common factors /causes 🡪 how do tests/items cluster?
* Measurement development; reliability, validity, diagnostic, monitoring, testing.
* To correct for measurement error
* To estimate factor scores (composite scores).

**Example of factor analysis:** test scores (Holzinger & Swineford, 1939).



* The y’s in the factor model are generally tests, scale-scores, or (survey-)items.
* The indicators are six tests:

visual perception - cubes - lozenges - paragraph comprehension - sentence completion - word meaning

Cubes, for example, consists of the total score of 32 timed problems to determine if two cubes are the same.

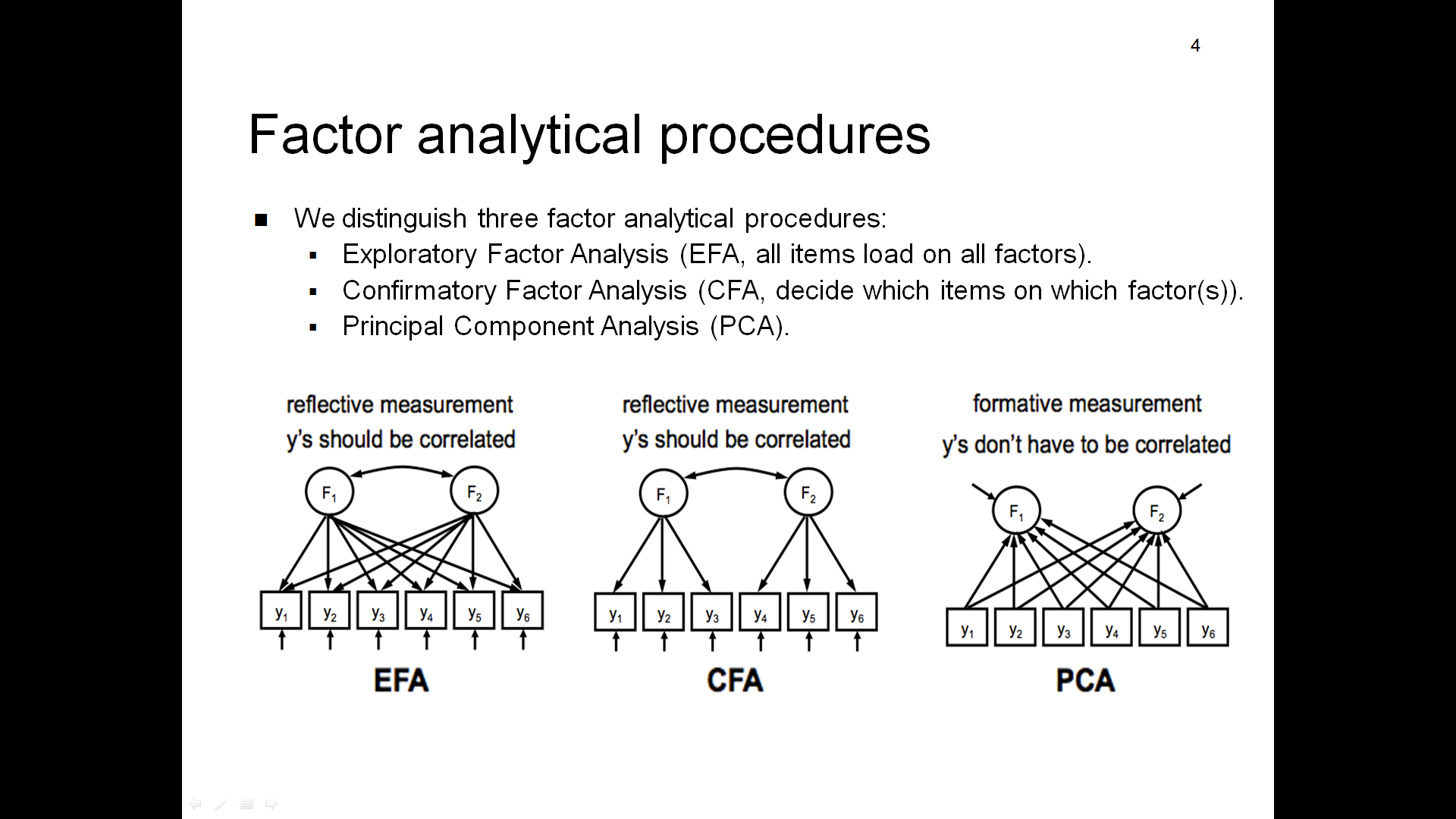
* It is hypothesized that the 6 tests measure two common factors: visual perception and verbal ability.

**The factor model**

* What’s in the model?
  + F is the **latent variable (unobserved): ALWAYS IN CIRCLE**
  + ε is the error term (unobserved).
  + y is the **manifest variable (observed): ALWAYS IN SQUARE**
  + λ is the factor loading.
  + θε is the variance of the error term.
* It is assumed that:
  + ρ(F,εi) = 0, E(εi) = 0
  + Variables have a multivariate normal distribution
* The model is specified by two equations:
  + y1 = λ1F1 + ε1
  + y2 = λ2F1 + ε2
* If the ε’s contain only random measurement error, then:
  + ρ(εi,εj) = 0
  + λi2 is the reliability of yi

**Factor analytical procedures**

1. Exploratory Factor Analysis (EFA, all items load on all factors).
2. Confirmatory Factor Analysis (CFA, decide which items on which factor(s)).
3. Principal Component Analysis (PCA).



**CFA: estimation & testing**

* Parameters estimated by minimization of the residuals, e.g. using Maximum Likelihood (ML)
* Identification – degrees of freedom = number of elements in the matrix – number of parameters.

Number of elements in the matrix = 1/2k(k+1) where k = nr of manifest variables

Number of parameters =

You must define or fix the scale of the latent variable:

* + - UVI: fix variance of latent var to 1 (scale is now z-score based)
    - ULI: fix the (first) factor loading to 1 (scale of lat var become equal to item’s scale)

**Measurement validity**

* Goal of factor analysis is to find ≥1 common factors.
* *a factor is just a mathematical abstraction* (Spearman)
* to give meaning to the factor, establish a network of lawful relationships each factor has with other factors:

*a scientific theory.*

* constructs receive their meaning from the deductive theory in which they are embedded. Postulation: there should be some a priori definition of the construct in the context of a relevant theory.
* **Validation:** the process of giving meaning to factors
* Tests of **construct validity** embed constructs in theory (hypotheses).
* Based on relationships with other variables: predictive, concurrent, criterion, convergent, discriminant validity:

All can be tested in the presence of a theory when data are available.

* **Content validity**: depends on extent to which empirical measurement reflects a specific domain of content

*e.g. an arithmetics test is not valid if it includes addition but not subtraction, multiplication and division.*

* **Face validity:** logical link between items and purpose, makes sense on the surface.

**Traditional statistics** all always have df = 0 in their SEM equivalents!

Are about explaining (differences in) scores or means: (*SEM contains all of these)*

* + **t-test** 🡪 differences in means
  + **ANOVA** 🡪 effect of an intervention/treatment
  + **Repeated measures ANOVA** 🡪 change over time
  + **Regression** 🡪 how well a variable can be predicted
  + **Principal component analysis** 🡪 determine the number of components

*“Relationships are everywhere, correlations are all around. “* And it’s up to us to explain these relations.

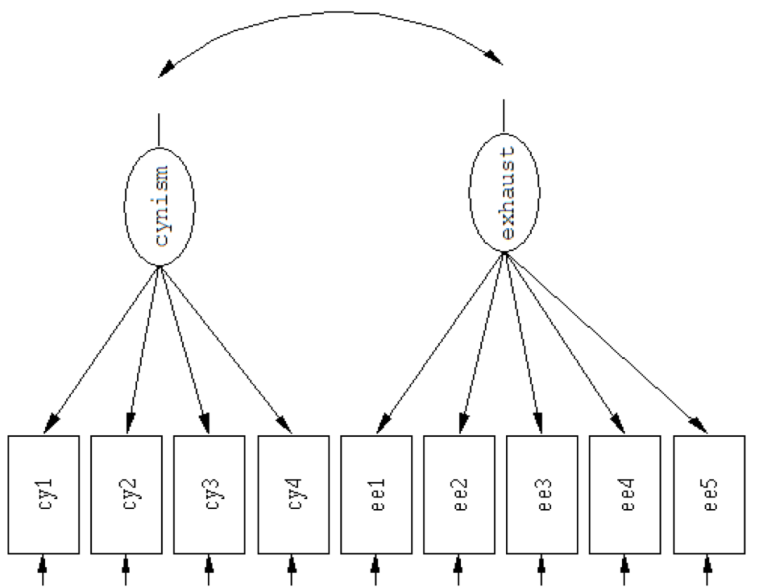
**Path model**

Visualization of set of hypotheses (effects, βs), together representing a theory (the path model): there is no single dependent variable!

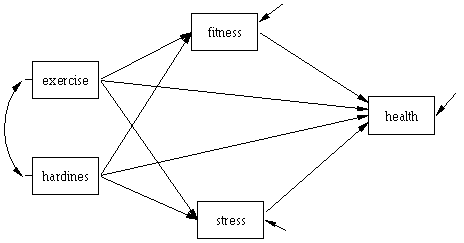
**Structural Equation Modeling – SEM**

* Method to visualize, estimate, and test a network of relations between variables:
* arrows/effects = hypotheses

🡪 z-test > 1.96

* model = theory, which explains:
  + correlations between variables
  + variances of the variables
  + variable means

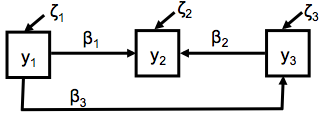
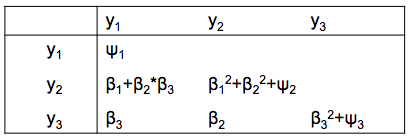
🡪 CHI2, p-value



**Decomposition rules:**

* the correlation (ρ) between two variables is equal to the sum of (1) the direct effect, (2) the indirect effects, (3) confounding effects, and (4) joint effects between these variables.
* Each indirect, confounding, and joint effect is equal to the products of the coefficients along the path going from one variable to the other while one can not pass the same variable twice and can not go against the direction of the arrows.
* The total variance of an endogenous variable is equal to the sum of (1) the amount of variance explained by the causal variables of this endogenous variable, and (2) the amount of unexplained variance.
* **! change the model = change how the relations between the variables are explained!**

|  |  |  |  |
| --- | --- | --- | --- |
|  | V1 | V2 | V3 |
| V1 | 1.00 |  |  |
| V2 | β1 | 1.00 |  |
| V3 | β1\*β2 | β2 | 1.00 |



**Estimation in SEM:**

SEM enables finding the population values for the model parameters (**θ**), but only works for **Σ**, the population correlation matrix: you always analyze the correlation matrix **S**, and **S** ≈ **Σ**.

That’s why you need estimators that are consistent, unbiased, and efficient:

* + **Consistency**: when the sample size gets larger (ad infinitum) then the probability to find the population value must also get larger.
  + **Unbiasedness**: the expected value of the estimator over many samples (ad infinitum) of the same size is the population value.
  + **Efficiency**: the variance (or the standard error) of the estimator is as small as possible.

In SEM there are many estimators available:

“ML” maximum likelihood. Continuous data only. Default.

**sample size**: 200-500, min 10cases/parameter 🡪 more needed when:

* + - R2 is smaller
    - collinearity is greater
    - ratio indicators : factor is smaller
    - non-normality of data is greater

**nonnormality i**s a threat generally when skewness > 2 and kurtosis > 7.

**categorical data** with >3 categories acceptable if distribution is normal.

“GLS” generalized least squares. Complete data only

“DWLS” diagonally weighted least squares

“ULS” unweighted least squares

“MLM” ML estimation w. robust SEs and a Satorra-Bentler scaled test statistic. Complete data only.

“MLMVS” ML estimation w. robust SEs and a mean- and variance adjusted test statistic. Complete data only.

“MLF” ML estimation w. SEs based on first-order derivatives and a conventional test statistic. Incomple data is OK!

“MLR” ML estimation w. robust (Huber-White) SEs and a scald test statistic. Incomplete data is OK!

Test statistic is (asymptotically) equal to the Yuan-Bentler test.

ML & ULS are both asymptotically consistent and unbiased, but ULSE is NOT EFFICIENT! So you better use ML.

**Testing in SEM**

* + Residuals can be non-zero due to sampling fluctuations (sample ≈ population) or misspecifications (incorrect model).
  + **Misspecifications due to either or both:**
    - **Over-parametrization**: parameters you estimate are actually zero in the population

*Check z-value*

* + - **Under-parametrization**: parameters you do not estimate (fixed to 0) are nonzero in the population

*Check modification index - MI*

* + When p >.05, the model is judged correct, and there is a 5% chance to have a Type I error (H0 is correct but rejected).

The **DFs of the CHI2 test** differs between models, and does not depend on sample size but on:

* + - the nr of observed variables in the model (A)
    - the nr of parameters that have to be estimated (B)

**df = A – B**

**A = n(n+1)/2**

* + If the model is rejected don’t evaluate the parameters (hypotheses).

**Modification index 🡪 look at when CHI2 high & significant but some goodness of fit indices indicate good model fit**

Parameters are estimated and fixed: when a “misspecified” parameter is estimated (in a hidden procedure) this will result in: **expected parameter change - EPC**: value for parameter

**modification index - MI**: decrease in CHI2 value.

*When MI > 3.84, introducing parameter to model will lead to a significant change in model fit*

Ask for by: summary(fit, …., modindices = TRUE, …)

**Assumptions ML Estimation**

1. **Observed variables are multivariate normally distributed**
   * + Structural models 🡪 only for the endogenous variables.
     + Factor models 🡪 for all indicators.

*If this assumption is violated the estimates are still consistent, but not efficient any longer.*

1. **Observed variables are observed without (measurement) error**.
   * + Structural models 🡪 for all variables.
     + Factor models 🡪 indicators may be observed with error.

*If this assumption is violated, estimates are biased, not consistent, and not efficient. Possible solution: correct for measurement error, use CFA.*

**Levels of missingness**

* + **item**(s) or cells *i.e. leaving an item or two blank from multiple item scales*
  + **scale**(s) or variables *i.e. omitting answers for an entire scale or construct*
  + **survey**(s) or cases *i.e. participants fail to complete an entire survey*

**🡪can be addressed sing systematic nonresponse parameters: SNPs**

For each level, but generally for scale level you can look at proportion/prevalence and patterns of missingness.

**Patterns of missingness**

* + MCAR missingness completely at random
  + MAR missingness at random
  + MNAR missingness not at random: problem but nothing you can do except change population

**Problems caused by missing data**

1. **Questionable external validity: response rate bias**

Results from the subsample may not be identical to those obtained from the entire sample

1. **Low statistical power**

Even when there is a true nonzero effect, sample too small to yield statistically significant result (Type II error) 🡪 smaller samples result in low statistical power, not missingness per se!

1. **Potentially invalid conclusions!**

**RULE OF THUMB: <5% of missing cases (not cells!) is ok, more should be investigated.**

**Ways to handle missingness:**

* + Listwise deletion: ok but you lose efficiency! % missing cells ≠ % missing cases!
  + Pairwise deletion: unwise, correlation matrix is often not positive definite
  + Single imputation – mean substitution, LOCF: unwise, biased estimates
  + Multiple imputation: good, but yields different results every analysis
  + **FIML full information maximum likelihood: best, analyze raw data**

Imputation strategies: single vs. multiple

* + **Single imputation**, e.g. mean imputation

One plausible value is imputed where data are missing

*Each value is randomly drawn from a conditional distribution, but there is no adjustment based on the variation in imputed values.*

* + **Multiple imputation**, e.g., maximum likelihood

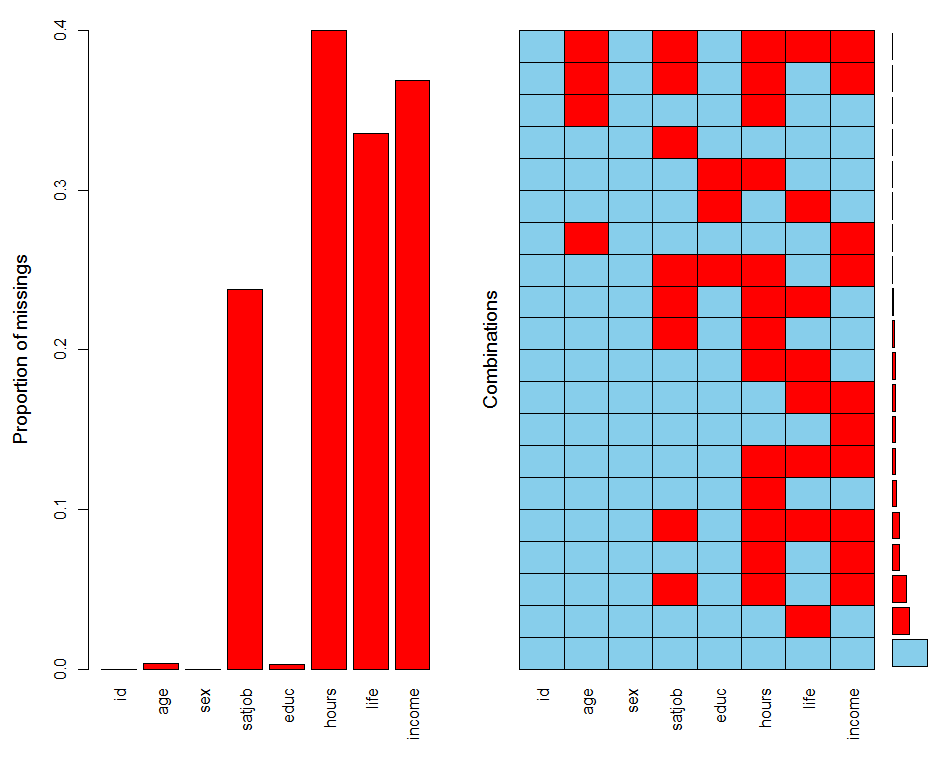
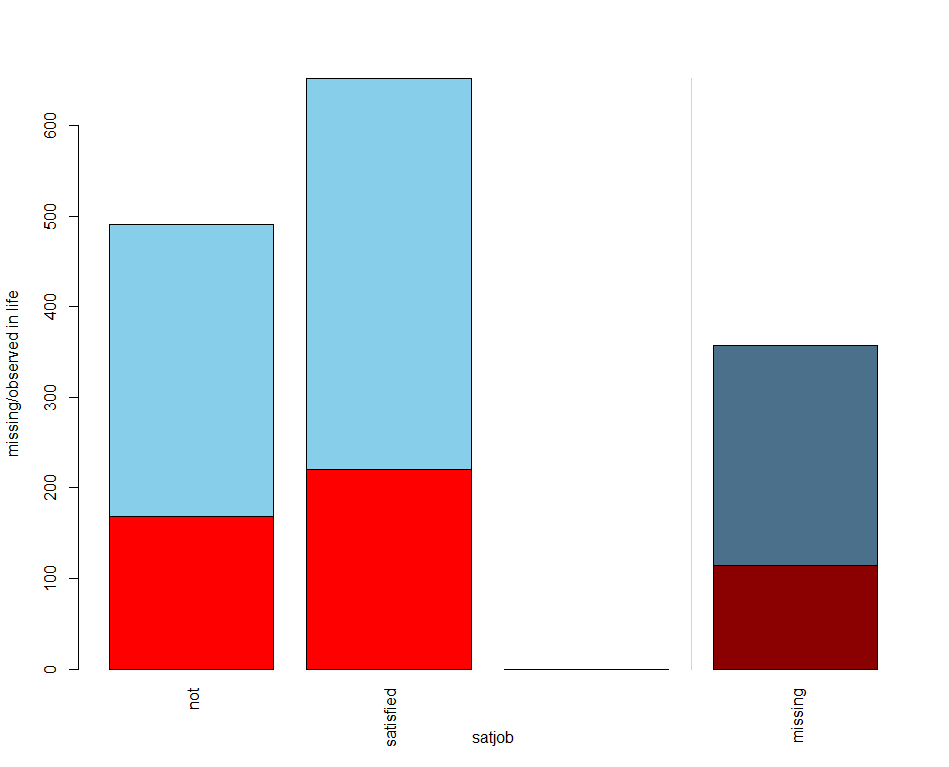
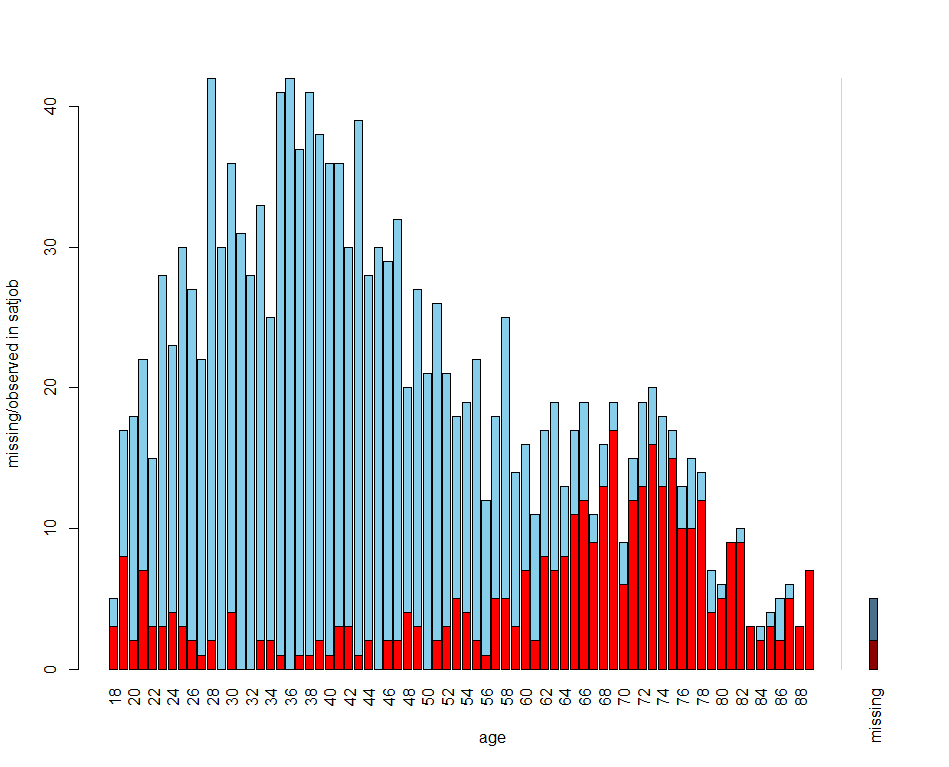
Several (or all) plausible values are considered, and estimates using different imputed values are pooled.

*The variation between the parameter estimates from different imputations is an estimate of imputation variance. Thus, MI is capable of disentangling sampling and imputation variance and providing an estimate of the total variance of the parameter estimates.*

**VIM package: Visualization and Imputation of Missing values**

a<- aggr(Data) b<- Data[ ,c(‘var1’, ‘var2’)] d<- Data[ ,c(“contvar”,”catvar”)]

a barMiss(b) histMiss(d)

**  **

**BaylorEdPsych 🡪 Little’s MCAR test: to test if missingness is MCAR**

Also install package mvnmle!

LittleMCAR(Data) or MCAR <- LittleMCAR(Data)

MCAR$chi.square [1] 812.0463

MCAR$df [1] 106

MCAR$p.value **[1] 0 🡪 missingness is not MCAR...impute!**

**Degrees of freedom**

The **DFs of the CHI2 test** differs between models, and does not depend on sample size but on:

* + - the nr of observed variables in the model (A)
    - the nr of parameters that have to be estimated (B)

**df = A – B**

**A = n(n+1)/2**

**df < 0** no solution, model “not identified”

**df = 0** 1 single solution for each model parameter: CHI2 = 0, *p* = 1.00, residuals = 0, model “saturated.

Cannot test how well correlations between variables are explained

Can test how well an x-variable explains a y-variable

**df > 0** Multiple but 1 best solution for each parameter, found by minimizing F. Typically *p* <1.00, residuals ≠ 0

Can test how well correlations between variables are explained

Can test how well x-variable explains y-variable, but only is *p*≥.05 (at *p* <.05, the model is rejected).

*Even when df ≥ 0, a model might not be identified because the sufficient conditions for identification are not fulfilled*.

**Lavaan** “it’s just right”

Lavaan provides the flexibility needed to handle most modeling issues you will come across.

Its basic structure:

1. model specification
2. model fitting to observed data
3. info summary of fitted model

Using 4 formula types, different lavaan operators, a large variety of latent variable models can be described!

|  |  |  |
| --- | --- | --- |
| **Formula type** | **operator** | **mnemonic** |
| Latent variable definition | =~ | Is measured by |
| regression | ~ | Is regressed on |
| (residual) (co)variance | ~~ | Is correlated with |
| intercept | ~1 | intercept |
| Defined parameter | := | Is defined as |
| Equality constraint | == | Is equal to |
| Inequality constraint | < | Is smaller than |
| Inequality constraint | > | Is larger than |

**Fitting functions**

* + **sem()** path analyses and SEM
  + **cfa()** confirmatory factor analysis
  + **growth()** latent growth curve modeling
  + **lavaan()** can be used for all models, but does NOT include default parameters!!

**Fit options**

* **Specify estimator and imputation strategy**

fit <- sem(model, data = Data, estimator = “MLR”)

fit <- sem(model, data = Data, missing = “ML”)

* **Add intercepts**

fit <- sem(model, data = Data, meanstructure = TRUE)

* **Use bootstrapping**
  + **Bootstrapped SEs and *p*-values**

fit <- sem(model, data = Data, se = ‘bootstrap’)

fit <- sem(model, data = Data, test = ‘bootstrap’)

**Summary options**

* Add standardized estimates: summary(fit, standardized = TRUE)
* Add GOF indices: summary(fit, fit.measures = TRUE)
* Add modification indices: summary(fit, modindices = TRUE)

**Goodness of fit measures**

summary(fit, fit.measures = TRUE) fitMeasures(fit)

* **Absolute fit:** compares model fit to no model at all
  + **CHI2** dependent on sample size and model complexity: bad!
  + **RMSEA**  root mean square error of approximation. Parsimony adjusted. Range 0-1

☺<.05

* + **GFI** goodness of fit index. Independent of sample size.

☺ >.09

* + **AGFI** adjusted (for parsimony) goodness of fit index. Independent of sample size.

☺ >.09

* + **RMR** root mean square residual: measure of mean absolute value of covariance resid

☺ = 0 or close to 0: 0 = perfect fit

* + **SRMR**  standardized root mean square residual: range 0-1

☺ <.05

* **Incremental fit:** compares model fit to a baseline model
  + NFI
  + NNFI
  + CFI comparative fit index:. Range 0-1

☺ >.95

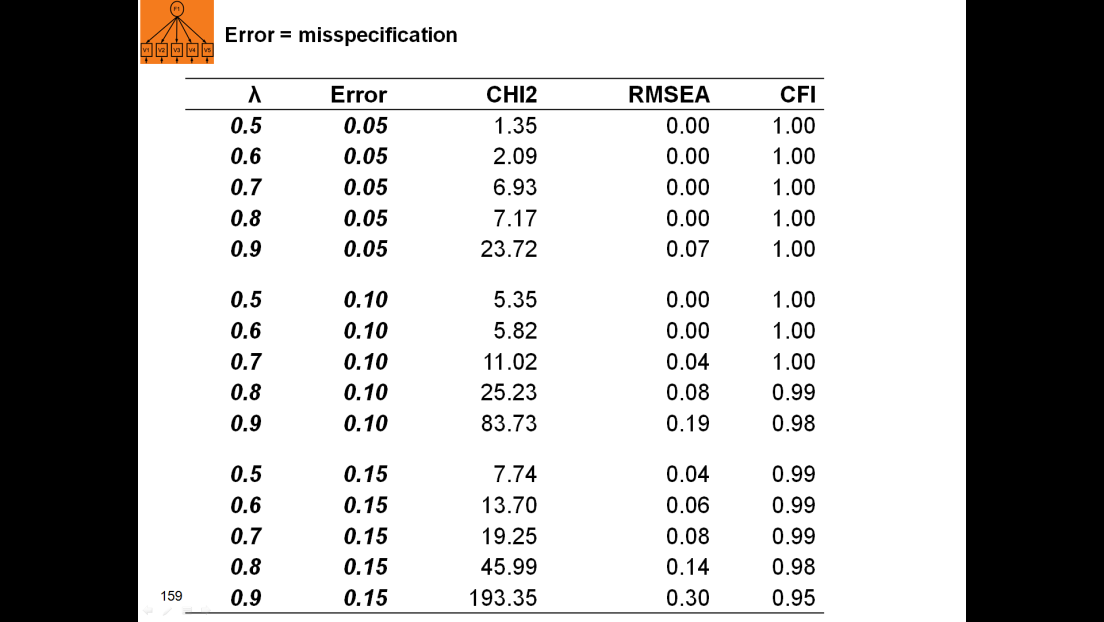
Which ones to report? Opinions differ, but suggested are: Chi, SRMR, RMSEA, CFI

**Modification indices (mi)**

summary(fit, modindices = TRUE)

To identify local misspecifications. In lavaan the mi are supplemented by the expected parameter change (epc) vales, of which it is best to look at the ones based on the model when all variables are standardized (not just the latent or observed)

*In general, the goal is to find a model that is “more or less” correct. Models are always simplifications of reality and are therefore always misspecified. It is possible to detect large misspecifications and ignore small ones, but also the other way around!*

Larger misspecifications should correspond to larger CHI2 & RMSEA >.05, and smaller CFI <.95. The same size misspecification should lead to almost (sampling fluct.) the same size CHI2, RMSEA, and CFI.

e.g. see left.

The extent of **model** **misspecification** is indicated by the **noncentrality parameter – ncp**. The larger the ncp, the larger the power to detect the misspecification. The larger the misspecification, the more noncentral the CHI2.

* + **ncp** = (MI/EPC2)δ2

high power: >.80

large misspecifications: δ ≥ .10

**Lavaan: a path model example**

install.packages(‘lavaan’, dependencies = TRUE) *install*

library(lavaan) *load*

pathex <- read.csv2("popANDprob.csv")  *import data*

*Personality and popularity as predictors of problems*

**model <-’**cbclDQ ~ popular + temper + sex

cbclAB ~ popular + temper + sex

cbclAD ~ popular + temper + sex‘

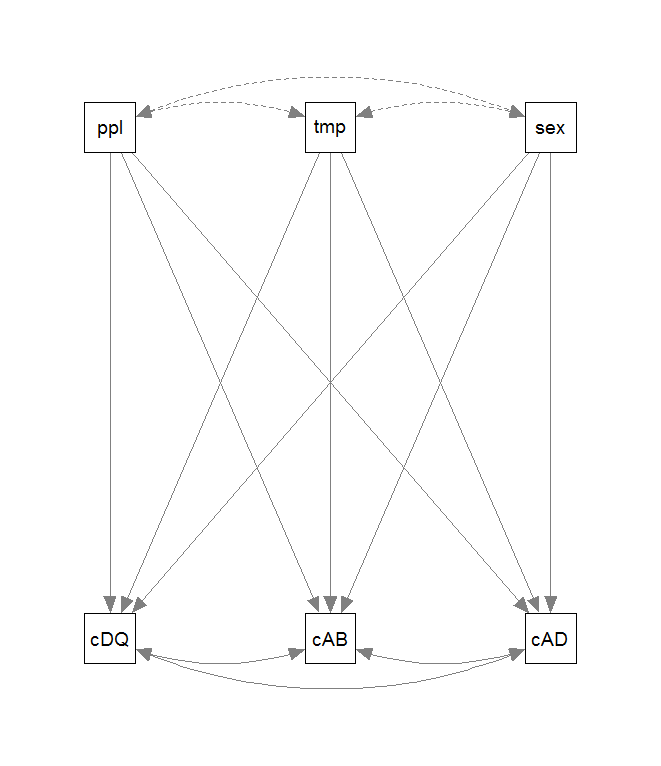
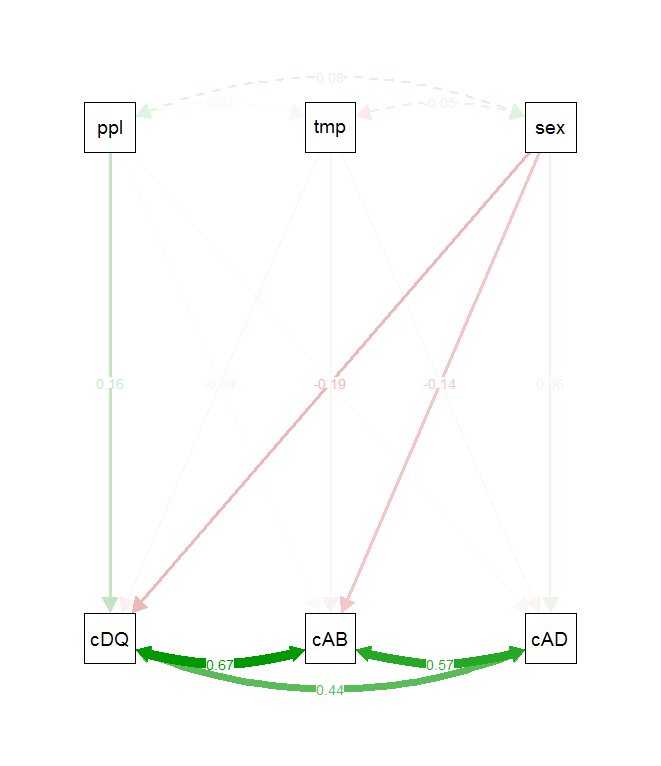
**fit** <- sem(model, data = pathex)

**summary**(fit, standardized = T)

*will give: parameter estimates (+SE, z value,* p*, std. lv, std.all)for the regressions, covariances, variances, whatever you put in the model.*

**semPlot**

semPaths(fit, residuals = F) vs. semPaths(fit, what = “std”, residuals = F)

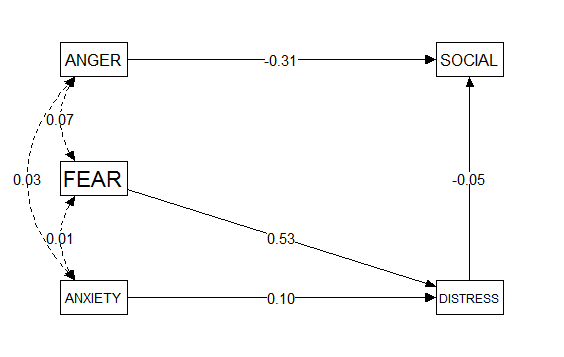
 

You can ask for a loooot of functions, e.g.:

semPaths(fit\_final, what= "col", "std", layout = "tree", rotation = 2,

intercepts = F, residuals = F, curve = 2, nCharNodes = 0,

edge.label.cex = 1, edge.color = "black", sizeMan = 10, sizeMan2 = 5)



**Model evaluation, improvement and CFA**

🡪 Omnibus testing of the model (global):

* CHI2 test statistic that is dependent on sample size and model complexity: bad!

*In general, model fit is sensitive to: sample size, nr of variables, non-normality, model characteristics, etc*

* GOF indices independent of sample size and model complexity: better to use!

🡪 Identifying model misspecifications (local): modification indices

SEM: when regression met factor analysis.

**Measurement Model (CFA):**

* The mapping of observed measures onto theoretical constructs
* Constituent parts: loadings of measures theoretical constructs, error vars, error covars (correlated errors)

**Structural Model (Regression):**

* The regression and correlational links between constructs
* Constituent parts: regression paths, variances of the exogenous variables, covariances between exogenous variables, variances of the disturbances of endogenous variables, covariances between disturbances, covariances between disturbances and exogenous variables (usually set to zero)

**Defaults of cfa()**

1. The factor loading of the first indicator of a latent variable is fixed to 1, thereby fixing the scale of the latent variable.
2. Residual variances are added automatically.
3. All exogenous latent variables are correlated.

**CFA example cfa()**

HS.model <- ' visual =~ x1 + x2 + x3 textual =~ x4 + x5 + x6 *# specify the model*

speed =~ x7 + x8 + x9 '

**fit <- cfa(**HS.model, data=HolzingerSwineford1939**)**  *# fit the model*

summary(fit, standardized = T, fit.measures=T) *# summarize output*

**CFA example lavaan()**

HS.model <- ' visual =~ 1\*x1 + x2 + x3 textual =~ 1\*x4 + x5 + x6 **# fix first loadings to 1**

speed =~ 1\*x7 + x8 + x9 '

x1~~x1 **# add (residual) variances (12)**

x2~~x2

x3~~x3

visual~~visual

visual~textual **#add covariances (3)**

**Practical issues in sem:**

Sample size: bigger is generally better, more reliable estimates

Missing values: patterns, mechanisms, robust algorithms

Basic assumptions: Multivariate normality

Absence of multicollinearity and singularity

Normality of residual variances (& resid covars should be small&symmetric)

**Data management**

Screen for outliers & missing values

Check univariate distributions and bivariate associations

* + - Skewness
    - Kurtosis
    - Singularity (extreme correlations)

Test multivariate assumptions prior to analysis (if possible), otherwise scrutinize available odel diagnostics (e.g., mod. Indices).

* + - Multivariate normality of covariance structures
    - Additional assumptions may apply!

**Trouble-shooting - how to deal with:**

Missingness 🡪 light imputation

Outliers 🡪 to winsorize, ignore or delete?

*Winsorization is the transformation of stats by limiting extreme valyes to reduce the effect of possible spurious outliers (e.g. for all outliers the value is changed to 2SD from the mean: you lose the order of extremity within the group of outliers!)*

*Especially viable when sample size <500 and only few outliers.*

*! proactive data collection strategies can reduce outliers and missingness!*

Lack of normality 🡪 weighted estimators, robust estimators, bootstrapping

**Missingness in repeated measures**

* Same three levels of missingness + time
  + Item, scale, & survey
  + Time = added complexity (level)…attrition, drop-out
  + **Patterns of missingness over time:**
  + Monotone (dropout)
    - when data are available at every assessment until a time when the participant drops out and provides no further assessments.
  + Intermittent
    - when there is a missing observation in between observed assessments.
  + Mixed
    - both monotone and intermittent missingness.
* The same mechanisms (MCAR, MAR, NMAR) apply to missingness in studies involving repeated measures
  + with the same consequences regarding bias and power

Longitudinal studies are sensitive to compounded (added) missingness!

**Assumptions of normality**

* Independent observations
* Large sample size
* Correctly specified model
* Multivariate normality
* Continuous data

If all conditions are satisfied...then “normal” maximum likelihood estimates (MLE) are accurate, efficient, and consistent

**Robust estimators of ULS, ML, &WLS are available.**

* + (..., estimator = “MLR”, ...)

**Test statistics and SEs can be easily obtained.**

* + (..., test = “Satorra-Bentler”, se = “robust.huber.white”)

**Bootstrapping is also easily accomplished.**

* + (..., test = “bootstrap”, se = “bootstrap”, ...)

**Bootstrapping**

* Resampling technique that treats the observed data as an estimate of the population.
  + - Bootstrapped samples are cases drawn with replacement from observed data.
    - Estimates from bootstraped samples form a distribution of statistic of interest
    - SEs calculated from bootstrapped estimates

Use when:

* Theoretical distribution of statistic of interest is complicated or unknown
* Sample size is insufficient for straightforward statistical inference
* Power calculations have to be performed, and a small pilot sample is available

**Total recap of slides so far:**

* You need to recognize the conditions that affect model fit:
  + Deviations from nonnormality
  + Model Complexity
  + Sample Size
  + Size of unrelated model parameters (in this case high factor loadings)
* Good models can be rejected and bad model can be accepted if any one of these conditions is at work.
* Should we therefore stop evaluating model fit with goodness of fit statistics? No. Accept that the world isn’t perfect:
  + know what sources affect your fit statistics (and parameter estimates),
  + learn whether any of those sources are present in your data,
  + assess (intuition) how much your results are affected.
  + use the CHI2, RMSEA, CFI to a lesser degree SRMR.
* SEM:
  + it is (very) easy to specify and estimate models.
  + It requires a lot of experience and knowledge to evaluate models.

**Moderation**

Interactions are used to test moderation:

* + Center all continuous measures, then create interaction
  + Remember to use proper coding scheme for any categorical predictors.
  + **In lavaan, the interactions need to be created and added to the data frame beforehand.**

That is, you cannot multiply predictors in model like in lm()

Multiple group analysis.

* If a categorical predictor represents independent groups, then SEMs can be estimated separately for each group.
* Analogous to “split file” in SPSS
* Differences between each parameter or groups of parameters can be tested to determine if groups differ.
* This is done with the use of equality constraints.

**Example**

model <- 'DISTRESS ~ ANXIETY + FEAR'

fit\_model <- sem(model, missing = "ML", data = Data, group = “gender”)

summary(fit\_model)

**single equality constraint**

model <- 'DISTRESS ~ c(a1,a1)\*ANXIETY + FEAR'

fit\_model <- sem(model, missing = "ML", data = Data, group = “gender”)

summary(fit\_model)

*The constraint forces the regression coefficients (not the weights) to be identical, so the model now has 1 df.*

**Multiple constraints**

ml4<-'DISTRESS ~ ANXIETY + FEAR'

fit\_model <- sem(model, missing = "ML", data = Data, group = "gender“, group.equal = c(“regressions”))

summary(fit\_model)

**What can be constrained – the group.equal argument allows you to test:**

* “loadings”: factor loadings on the observed variables
* “intercepts”: intercepts of the observed variables
* “means”: intercepts/means of the latent variables
* “residuals”: residual variances of the observed variables
* “residual.covariances”: residual covariances of the observed variables
* ”lv.variances”: (residual) variances of the latent variables
* “lv.covariances”: (residual) covariances of the latent variables
* “regressions”: all regression coefficients in the model

In summary

* **Multiple groups analysis > moderation**
* Regression paths, factor loadings, etc.
* Use ΔΧ2 test (ΔCFI?)
* **Single or multiple parameters can be tested**
* Individual equality constraint: c(a1,a1)\*
* Multiple constraints: group.equal = c(“regressions”)

**Measurement invariance**

* **Configural** invariance
* Does the model fit the data separately for each group?
* **Metric** (weak) invariance
* Are the factor loadings the same for each group?
* **Scalar** (strong) invariance
* Are the loadings and intercepts the same for each group?
* **Strict** invariance
* Are the loadings, intercepts, and residual variances the same for each group?

**How to check measurement invariance:**

1. First run a **configural model**

* multiple groups are estimated without constraints

burnsex1 <-cfa(burn\_sex, data = burnout, group = "sex")

1. Run a **metric (weak) model**

* the **factor** **loadings** are constrained.

burnsex2 <-cfa(burn\_sex, data = burnout, group = "sex", group.equal = c("loadings"))

1. Run a **scalar (strong) model**

* loadings and **intercepts** are constrained to be equal (scalar invariance).

burnsex3 <-cfa(burn\_sex, data = burnout, group = "sex", group.equal = c("loadings", "intercepts"))

1. Run a **strict model**

* Loadings, intercepts AND **residual variances** are now constrained (fixed to be equal across groups)

burnsex4 <-cfa(burn\_sex, data = burnout, group = "sex", group.equal = c("loadings", "intercepts", "residuals"))

You will see that with increasing levels of these models, the degrees of freedom will also increase!

These models are **nested** so can be tested using chi-square difference test, i.e. ANOVA(model1, model2)

**measurementInvariance()** from the semTools package does this kind of automatically

fit <- measurementInvariance(model, data = Data, group = “groupingvariable”)

**Summary of measurement invariance**

* Testing invariance is important: you should not make some group comparisons
* Testing invariance can be complicated due to differences in terms, recommendations, and partial invariance
* Testing invariance with lavaan is easy, but does require some understanding of MI

**Psychometric CFA properties**

Steps in scale development:

1. Selection of the ‘best’ items on face validity.

Assessment of psychometric properties:

1. Perform factor analysis (validation of the dimensional structure).
2. Assess the scale-reliability, e.g. with Cronbach’s alpha.
3. Perform (construct) validity tests.

Another psychometric property is whether the scale ‘works’ equally well for different groups, e.g. patients with personality problems vs patients with anxiety problems (invariance testing, measurement equivalence).

**Measurement equivalence:** concerns the invariance of the measurement properties:

* Factor structure
* Validity
* Reliability

*If a measurement instrument is invariant you can use it to make comparisons between groups and/or over time.*

Most important is that responses on a specific question must be comparable between persons.

You need to be able to test whether people in different groups:

* express themselves on the same dimension (e..g., length),
* use the same metric (e.g., centimeters),
* use the same reference point (e.g., 0).

↓

**Same dimension: Configural Invariance**

* Estimated model: x(n) = Λ(n)F(n) + δ(n)
* Correct: same concept is measured in all groups by the instrument(s)

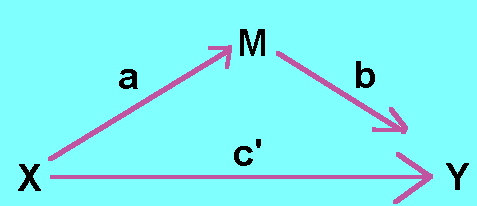
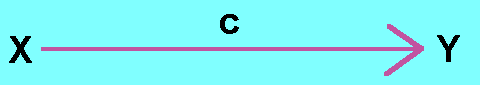
**Same metric: Metric Invariance**

* Estimated model: x(n) = Λ(n)F(n) + δ(n)
* Assumptions tested: Λ(1)= Λ(2)=…= Λ(n)
* Correct: scale-point distance is equal across groups: relationships between (metric invariant) variables are comparable across groups.

**Same reference point: Scalar Invariance**

* Estimated model: x(n) = τ(n) + Λ(n)F(n) + δ(n)
* Assumptions tested: previous + τ(1)= τ(2)=…= τ(n)
* Correct: zero-point on same place in all groups - the **means** of (scalar invariant)variables can be compared across groups (reference point).

**Mediation**

path c’ is the direct effect path c is the total effect

a and b are also direct effects

ab is the indirect (mediation) effect: it is defined as the reduction of the effect of X on Y (c – c’)

so, ab ≈ c – c’

but only when exactly the same participants are used in each of the regression equations!

**Complete mediation:** when X no longer affects Y at all! (c’= 0)

This rarely occurs in regression models.

**Partial mediation:** when X to Y is reduced in absolute size but is still different from zero.

You need to prove that X predicts M (test path a) and that M predicts Y (test path b). Optional is proving that X predicts Y (test path c). You can then show that M reduces the effect of X on Y (test c’ and ab).

**Bootstrapping** is the best way to do this!

**Mediation design issues**

* Proximal and distal mediation Mediator could be chosen too close in time to X or Y
* Reverse causal effects (alternative models) Does Y predict X? Does M predict X? Does Y predict M?
* Measurement error Latent variables provide advantages, as do highly reliable measures
* Omitted (3rd) variables Always an issue
* Moderated mediation/mediated moderation? See the website of Andrew Hayes

**Mediation in lavaan**

Indirect effects need to be defined!

* + - This is done using parameter labels (a, b, c)
    - Total effect can also be defined (c + a\*b)

model <- ' Y ~ c\*X # direct effect (path c’)

M ~ a\*X # mediator (path a)

Y ~ b\*M # mediator (path b)

ab := a\*b # indirect effect (path ab)

total := c + (a\*b) # total effect (path c)

and more complicated:

model <- ' Y ~ c1\*X # direct effect (path c’)

M1 ~ a1\*X # 1st mediator (path a)

M2 ~ a2\*X # 2nd mediator (path a)

Y ~ b1\*M1 # 1st mediator (path b)

Y ~ b2\*M2 # 2st mediator (path b)

ab1 := a1\*b1 # 1st indirect effect (path ab)

ab2 := a2\*b2 # 2nd indirect effect (path ab)

**Sometimes you use a correlation matrix as input**

matrix <- '

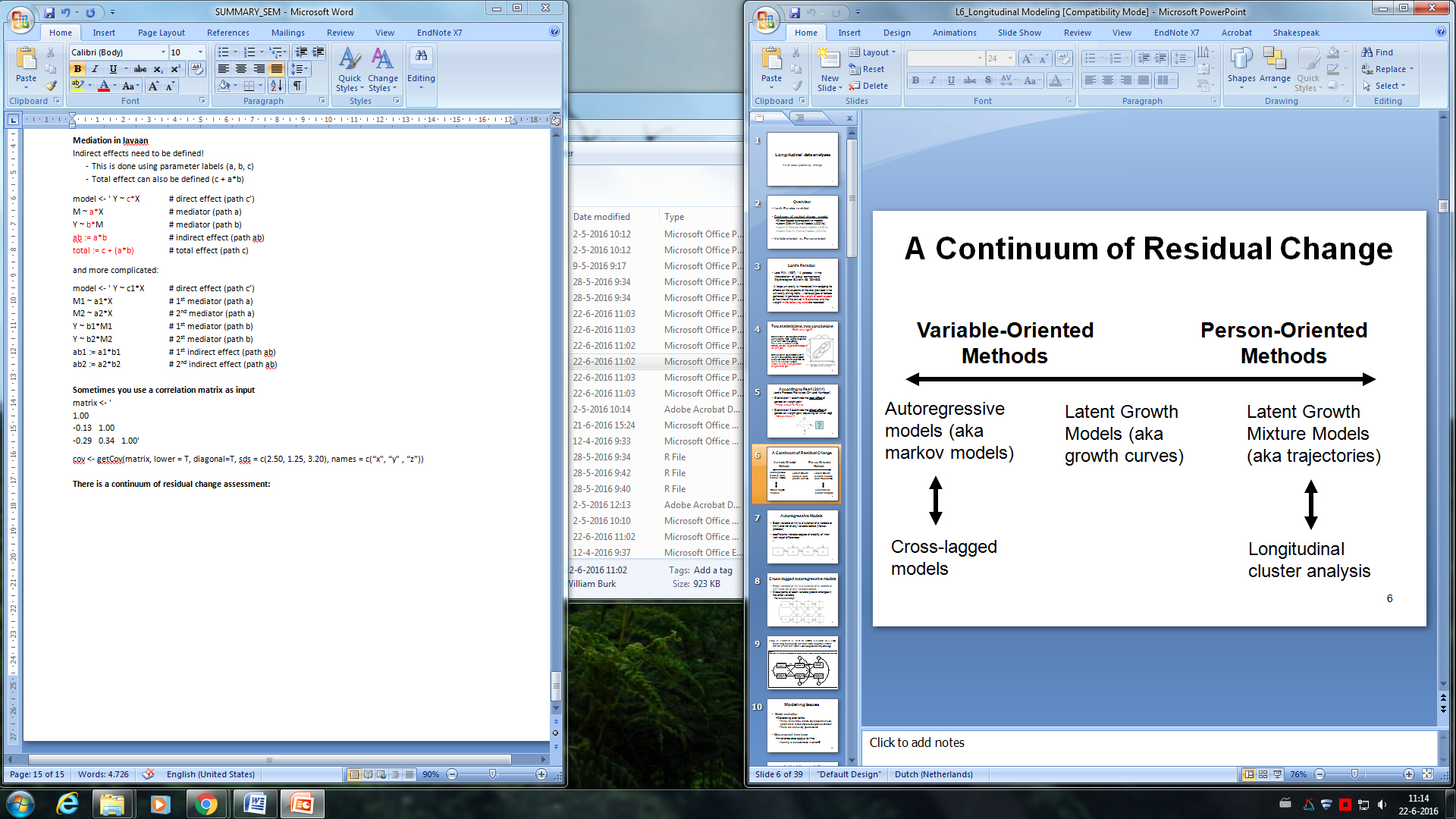
1.00

-0.13 1.00

-0.29 0.34 1.00'

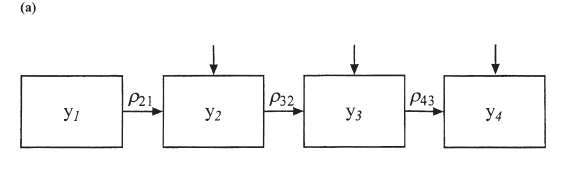
cov <- getCov(matrix, lower = T, diagonal=T, sds = c(2.50, 1.25, 3.20), names = c(“x", “y" , “z”))

**There is a continuum of residual change assessment:**

****

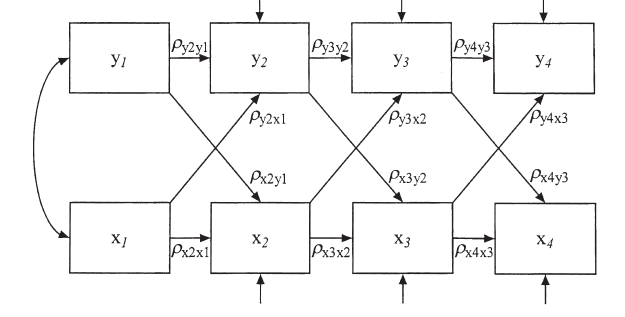
**Autoregressive models**

* Each variable at t(n) is a function of a variable at t(n-1) and not of any variable before (Markov process)
* Coefficients indicate degree of stability of inter-individual differences



**Cross-lagged autoregressive models**

* Each variable at t(n) is a function of a variable at t(n-1) and not of any variable before.
* Cross-paths of each variable predict changes in the other variable
  + Bi-directionality?



**Modeling issues**

* Model evaluation: correlating error terms. Is there any reason the model parameters reflect shared unexplained variance?
* Measurement invariance: also applies to time. (testing is less automated in lavaan), its not just for testing group differences! Loadings, intercepts, regressions, and variances can also be (in)variant across time!
* *To test this you need to constraint different paths to be equal (instead of constraining the same path to be equal across different groups).*

model <- ’t2 ~ **v1\*** t1

t3 ~ **v1\*** t2’

**example of cross-lagged autoregressive model specification**

model\_CL <- '

#stability paths

lonely2 ~ lonely1

lonely3 ~ lonely2

withdrawn2 ~ withdrawn1

withdrawn3 ~ withdrawn2

#crosspaths

lonely2 ~ withdrawn1

lonely3 ~ withdrawn2

withdrawn2 ~ lonely1

withdrawn3 ~ lonely2

#covariances

withdrawn1 ~~ lonely1

withdrawn2 ~~ lonely2

'

fit\_CL <- sem(model\_CL, estimator = "MLR",missing = "ML", data = longex)

summary(fit\_CL, fit.measures = T, standardized = T)

**LGCMs - latent growth curve models**

can be performed using mixed effects models/multilevel modeling, or with SEM. The random effects are captured by latent variables. **LGCMs require at least 3! time points for each individual**!

**Advantages of MEM or MLM vs. SEM**

Better with time-unstructured data Measures of absolute fit

Easier with many times Easier to respecify; more options for respecification

Better with fewer participants More flexibility in the error covariance structure

Easier with time-varying covariates Easier to specify changes in slope loadings over time

Random effects of time-varying covariates allowable Allows latent covariates

Allows missing data in covariates

**Key parameters of LGCM**

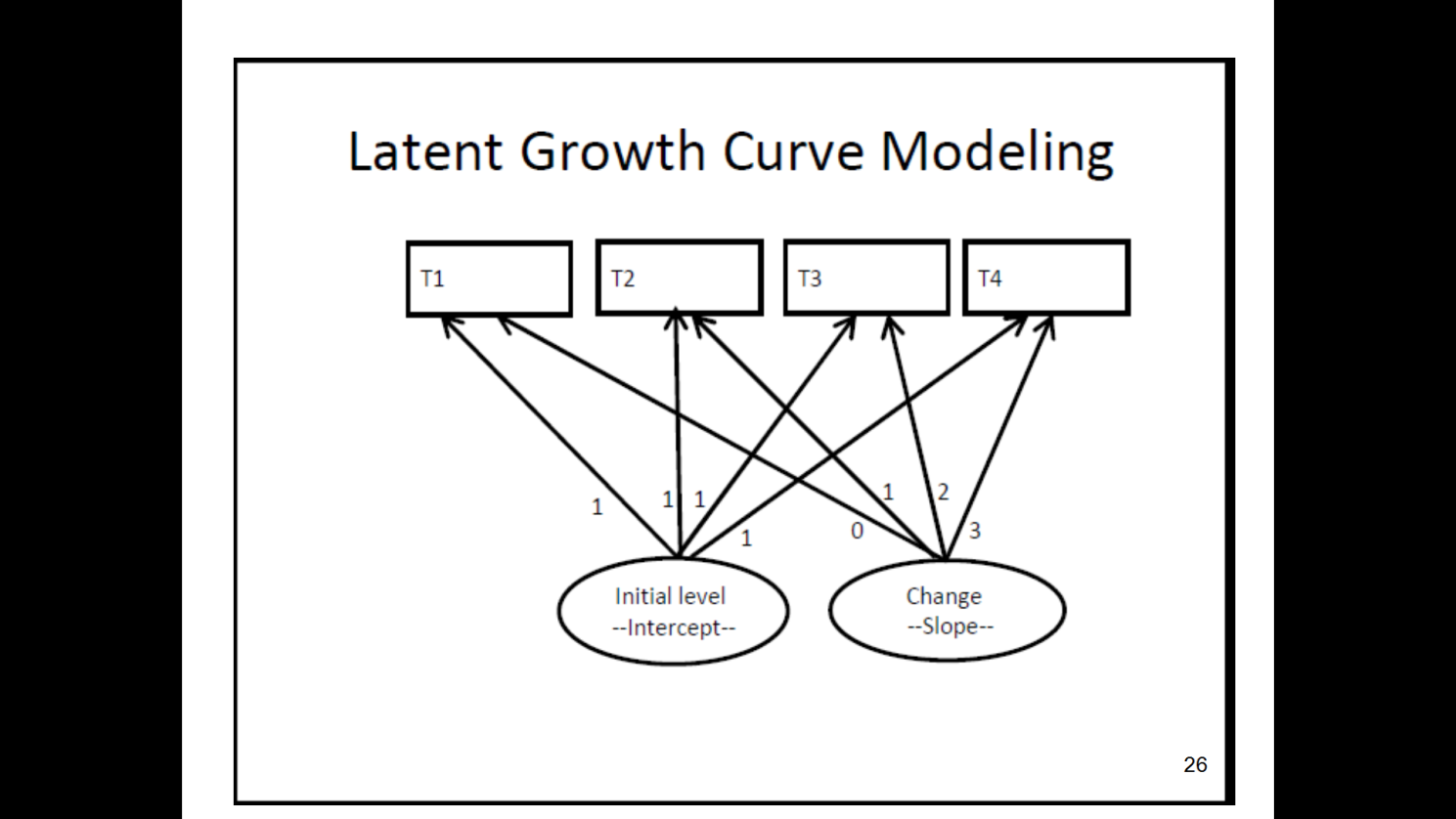
* **Error:** How far the score is from the line
* **Slope:** the rate of change
  + Some people are changing more than others and so have larger slopes.
  + Some people are improving or growing (positive slopes).
  + Some are declining (negative slopes).
  + Some are not changing (zero slopes).
* **Intercept:** where the person starts

*For both the slope and the intercept there is a mean & variance:*

* + **Mean**
    - Intercept: Where does the average person start?
    - Slope: What is the average rate of change?
  + **Variance**
    - Intercept: How much do individuals differ in where they start?
    - Slope: How much do individuals differ in their rates of change: “Different slopes for different folks.”

Model evaluation: don’t use CFI and TLI. **Never** have the intercept “cause” the slope factor or vice versa. Do not interpret standardized estimates except the slope-intercept correlation.

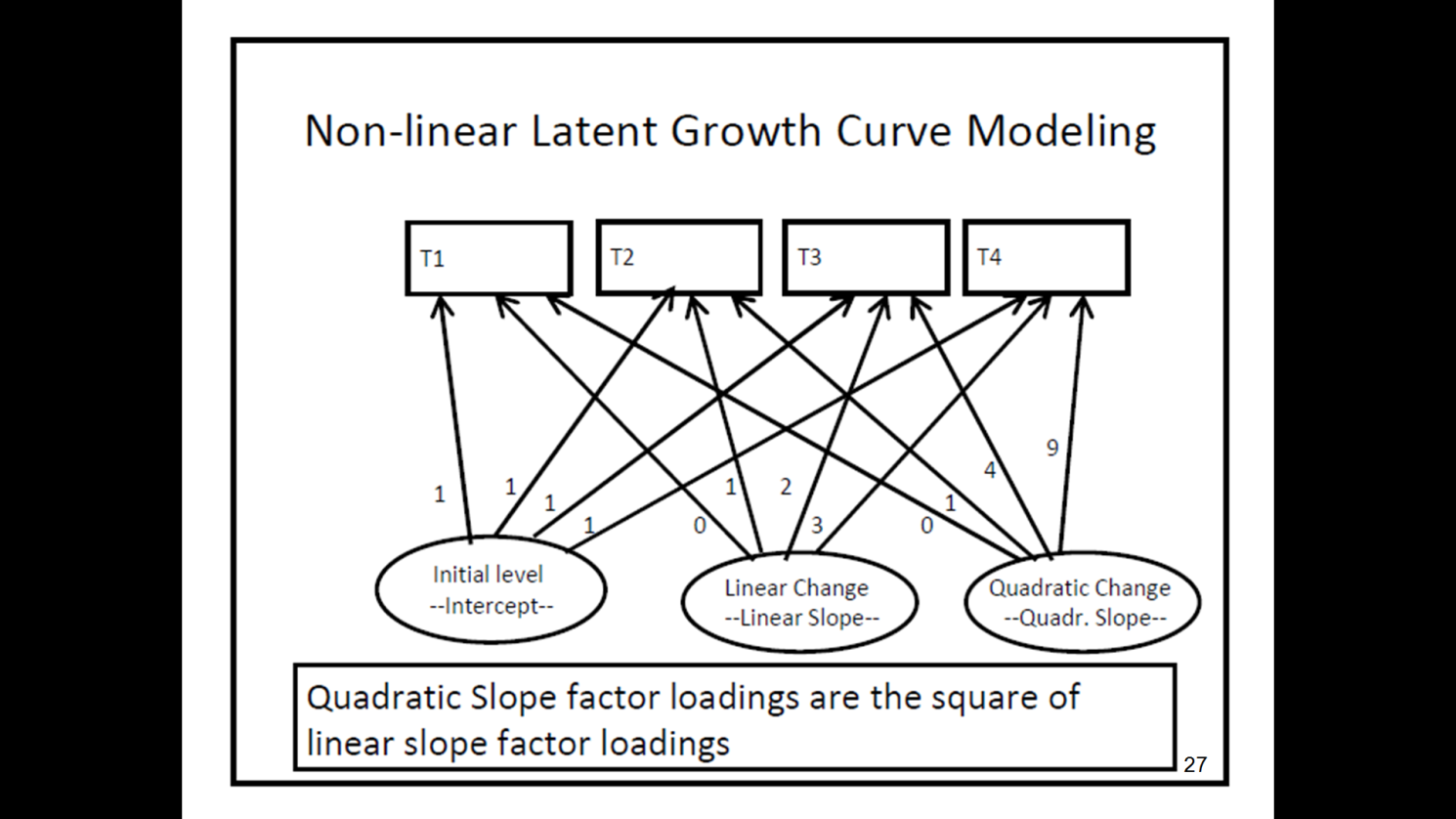
**Latent growth curve modeling**



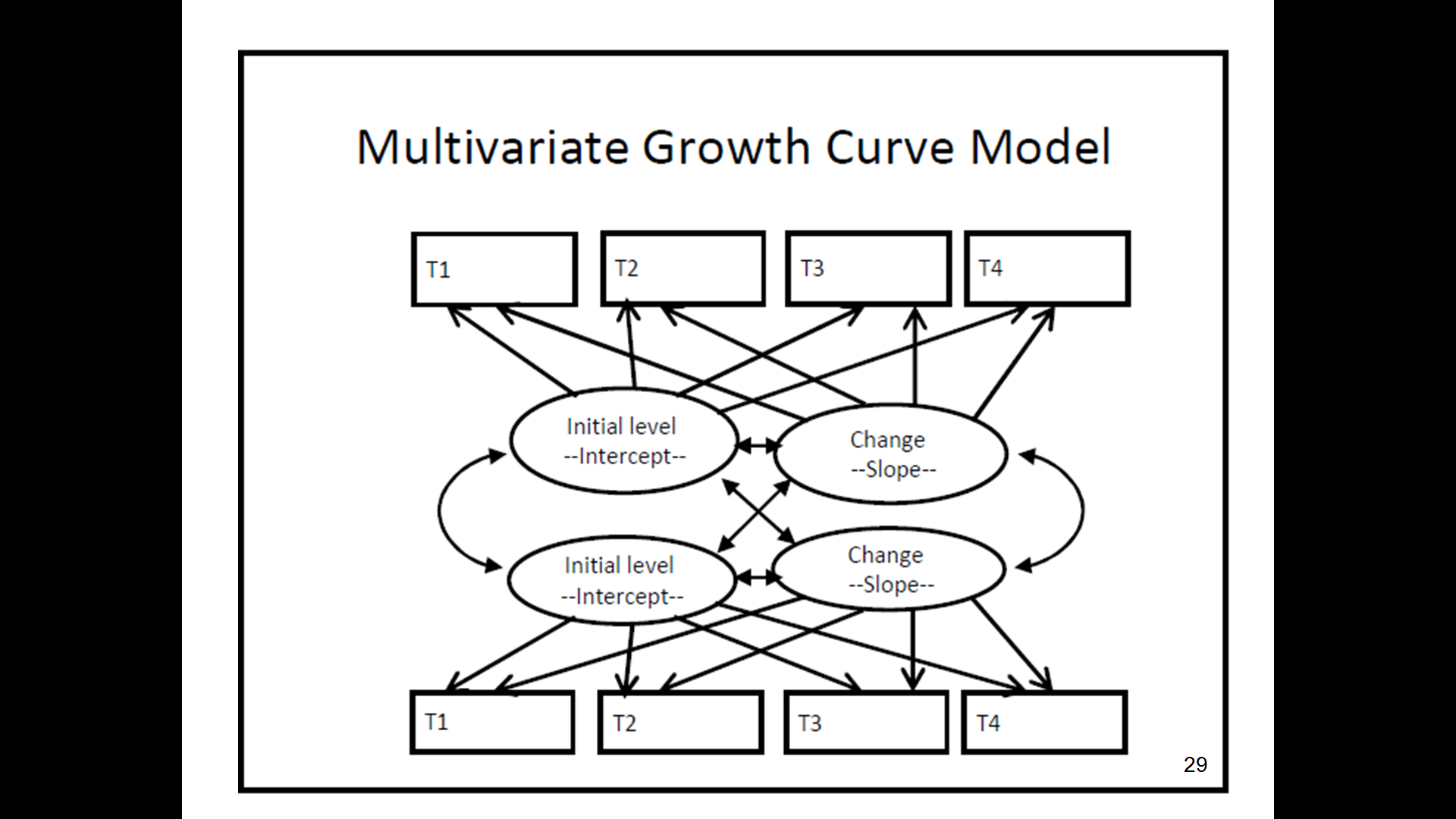
**Non-linear latent growth curve modeling**

Quadratic slope factor loadings are the square of linear slope factor loadings: (0, 1, 4, 9)

Logarithmic slope factors are log *e* (linear slope factor loadings + 1): (0, 1, 4, 9) are now (0, 0.69, 1.10, 1.39)

****

**Multivariate growth curve model**



**Example of multivariate LGCM specification**

model\_LGMplus <- '

# intercept and slope with fixed coefficients for loneliness

i\_l =~ 1\*lonely1 + 1\*lonely2 + 1\*lonely3

s\_l =~ 0\*lonely1 + 1\*lonely2 + 2\*lonely3

# intercept and slope with fixed coefficients for withdrawn

i\_w =~ 1\*withdrawn1 + 1\*withdrawn2 + 1\*withdrawn3

s\_w =~ 0\*withdrawn1 + 1\*withdrawn2 + 2\*withdrawn3

# regressions

s\_l ~ i\_w

s\_w ~ i\_l

i\_w ~~ i\_l

'

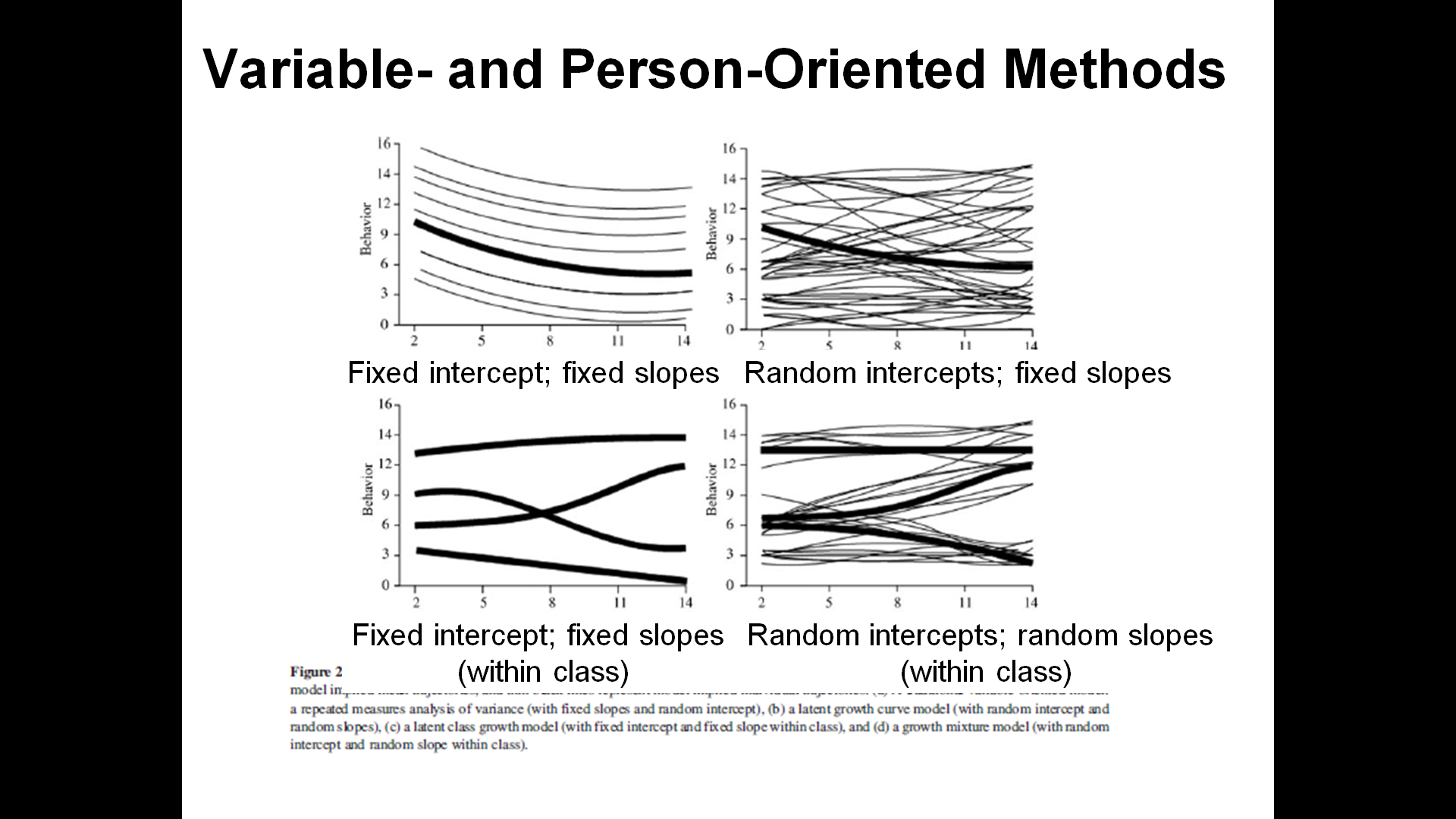
fit\_LGMplus <- growth(model\_LGMplus, estimator = "MLR",missing = "ML", data =longex)

summary(fit\_LGMplus, fit.measures = T)

Different models will have different results with different interpretations. You can inspect such different models based on:

* + - Distributions (transformations)
    - Testing constraints (cross-lagged)
    - Re-calculate GOF indices (LGCM)
    - Simplification/extension

**Variable- and person-oriented methods**

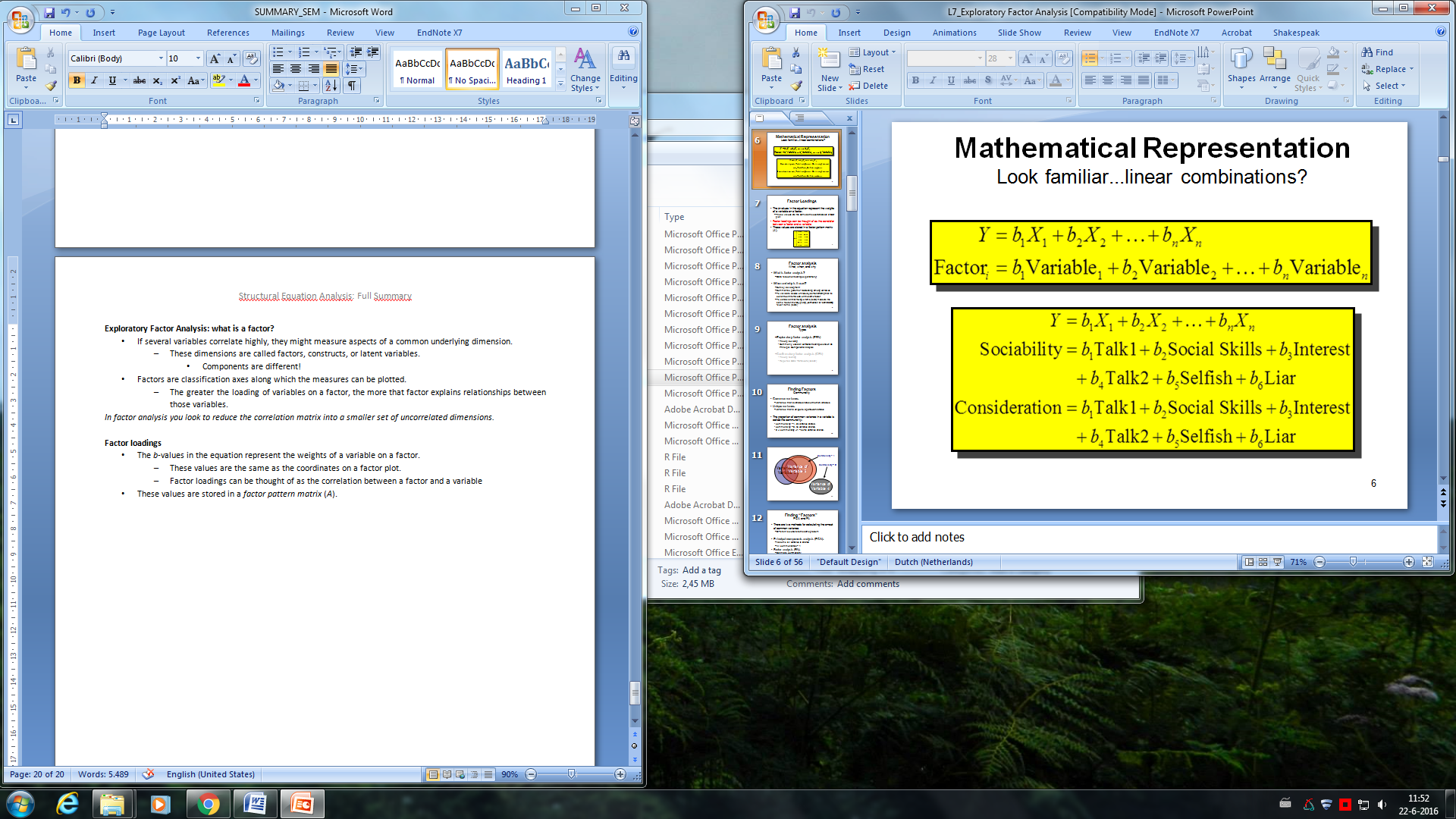


**Exploratory Factor Analysis**

Fatcor analysis is a data reduction technique (parsimony), used for survey-development, to summarize patterns of relationships among variables, to validate a scale or index by demonstrating that its constituents items load on the same factor, or to establish that multiple items (tests) measure the same factor, thereby giving justification for administering fewer items (tests).

* If several variables correlate highly, they might measure aspects of a common underlying dimension.
  + These dimensions are called factors, constructs, or latent variables.
    - Components are different!
* Factors are classification axes along which the measures can be plotted.
  + The greater the loading of variables on a factor, the more that factor explains relationships between those variables.

*In factor analysis you look to reduce the correlation matrix into a smaller set of uncorrelated dimensions.*



**Factor loadings**

* The *b*-values in the equation represent the weights of a variable on a factor.
  + These values are the same as the coordinates on a factor plot.
  + Factor loadings can be thought of as the correlation between a factor and a variable
* These values are stored in a *factor pattern matrix* (*A*).

**Exploratory factor analysis (EFA)**

* + - Theory building
    - Commonly uses an extraction technique known as Principal Components Analysis

**Confirmatory factor analysis (CFA)**

* + - Theory testing
    - Requires SEM framework (lavaan)

**Communality**

Common variance:

* + Variance that a variable shares with other variables.

Unique variance:

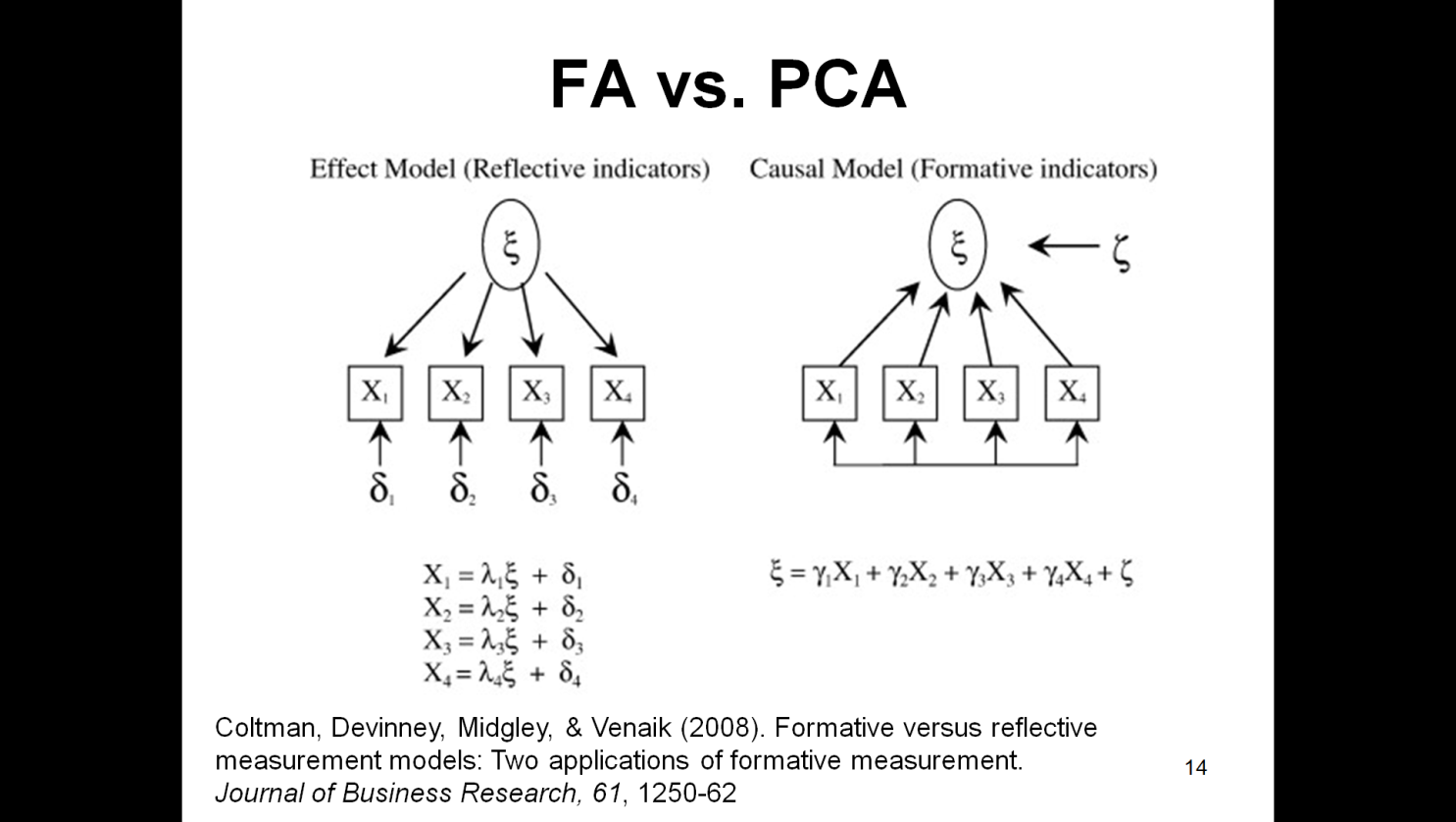
* + Variance that is unique to a particular variable.

The proportion of common variance in a variable is called the *communality*.

* + communality = 1, all variance shared.
  + communality = 0, no variance shared.
  + 0 < communality < 1 = some variance shared.

There are two methods for calculating the amount of common variance: different solutions to the same problem.

**Principal components analysis (PCA):**

* + Assume all variance is shared
  + All communalities = 1

**Factor analysis (FA):**

* + Estimate communality
  + Use squared multiple correlation (SMC)

**↓**

**Formative measurement (PCA)**

* + Items do not necessarily correlate
  + Items “formed” the latent construct
  + Example: Delinquency

**Reflective measurement (FA)**

* + Items are correlated
  + Items “reflect” the latent construct
  + Example: Self-esteem; personality

**Steps in EFA (PCA)**

1. Preliminary analyses
   * Identify sampling adequacy
   * Examine correlation matrix
2. Extract/Determine number of factors
3. Rotate factors/Interpret the results

**Initial considerations**

* **The quality of analysis depends upon the quality of the data (GI**⇒**GO).**
  + A factor solution will (almost) always be found, is the solution meaningful?
* **Sample size**
  + Factor analysis requires large sample sizes
    - “Rules of thumb” vary, depends on # of items and factors
    - T&F recommend minimum of 300

**Kaiser–Meyer–Olkin (KMO) measure of sampling adequacy:**

* + - squared correlation / squared partial correlation
    - should be greater than .5

**How to examine the correlation matrix**

* Variables should be positively correlated (*r* > .3), but avoid multicollinearity and singularity.
  + If all *r*s > .3, then consider using FA
  + If all *r*s < .3, then consider using PCA
* **Bartlett’s test of sphericity:**
  + Tests whether variables in matrix are correlated (i.e, whether the correlation matrix is an identity matrix)
  + should be significant at *p <* .05
* **Determinant:**
  + An indicator of multicollinearity
  + should be greater than 0.00001

**Factor extraction**

Extraction is the process by which factors are determined from a larger set of variables

* + Are multiple factor extraction methods
  + Principal components analysis is the most common extraction method for EFA
    - ML is commonly used for CFA

The goal is to extract factors that explain as much variance as possible with as few factors as possible- parsimony

There are different types of factor extraction techniques - PCA is the most common technique. However there are a variety of different techniques, each of which are appropriate in different situations. There are other techniques which may be more appropriate than PCA. When the purpose of your study is truly exploratory, PCA is the appropriate factor extraction technique. A note on terminology, **PCA actually extracts components as opposed to factors**. Components are **aggregates of variables**. There are no theoretical assumptions about the components causing the variables. Factors are underlying constructs that drive observed variables, as seen in SEM.

There are potentially as many factors as there are variables in your analysis. How many factors should you extract? This is a key question in factor analysis. The guidelines for how many factors should be extracted are known as stopping rules. Various stopping rules have been developed.

**Determining the nr of factors**

* **Kaiser’s criterion**
  + Kaiser (1960): retain factors with eigenvalues > 1.
  + Jolliffe (1972, 1986): retain factors with eigenvalues > .7

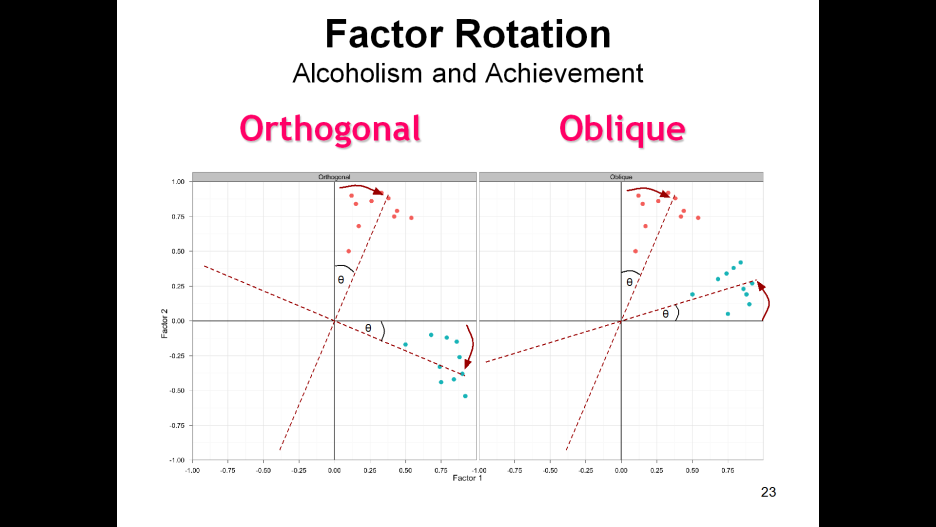
*Eigenvalue is a single value representing an eigenvector (= factor loadings)*

* **Scree plot**
  + Cattell (1966): use ‘point of inflexion’ of the scree plot.
  + Point of inflexion is error factor and is not retained.
* **Which stopping rule?...It depends** 
  + Check the communalities!
  + There are other options as well (e.g., parallel analysis)
    - Extracting factors with ML provides a GOF of factor solution

**Factor rotation**

To aid interpretation it is possible to maximize the loading of a variable on one factor while minimizing its loading on all other factors. There are 2 types:

* + **orthogonal** (factors are uncorrelated)
    - Varimax (preferred)
    - Quartimax (usually leads to many variables loading on one factor)
  + **oblique** (factors intercorrelate)
    - Promax (faster and designed for large data sets)
    - Oblimin



**Interpreting factor loadings**

Interpret rotated solutions!

* Ideally you want variables to load >.40 on one factor and <.30 on all other factors (unique loading)
* Generally exclude variables that load > .40 on more than one factor (cross-loading) and those that load <.30 on any factor

The “significance” of factor loadings depend on sample size (Stevens, 2002)

* n = 50, 100, 200, 300, 600, 1000
* loading = .72, .51, .36, .30, .21, .16

**R packages for EFA**

corpcor, GPArotation, psych (for fa() or principal()), pastecs

**Sampling adequacy and examining correlation matrix**

KMO measure (function created by G.J. Kerns)

kmo(raqData) $overall [1] 0.9302245

Calculate and test the correlation matrix:

raqMatrix<-cor(raqData)

cortest.bartlett(raqData) $chisq [1] 19334.49 $p.value [1] 0

$df [1] 253

det(cor(raqData)) [1] 0.0005271037

**factor extraction**

By extracting as many factors as there are variables we can inspect their eigenvalues and make decisions about which factors to extract. To create this model we use the principal() function:

pc1 <- principal(raqData, nfactors = 23, rotate = "none")

pc1

fa1<- fa(raqData, nfactors=23, rotate="none", fm="minres", oblique.scores = T)

**determining the number of factors**

plot(pc1$values, type = "b")



The first plot is the plot from the principal components analysis of RAQ data.

The second plot shows the point of inflexion at the fourth component.

Now you know how many components 🡪 extract 🡪 rerun the analysis, specifying that number:

pc2 <- principal(raqData, nfactors = 4, rotate = "none")

pc2

output will give h2, the communality, and u2, the unique variance for each factor.

The difference between the reproduced matrix and the actual correlation matrix is referred to as the residuals:

factor.residuals()

Provides the residual matrix, but we only need the values above (or below) the diagonal

residuals<-factor.residuals(raqMatrix, pc2$loadings)

residuals<-as.matrix(residuals[upper.tri(residuals)])

Now that the residuals are represented as a single column of data you can calculate:

**Number of large residuals** (those greater than .05)

large.resid<-abs(residuals) > 0.05

sum(large.resid)

**Percent of large residuals** (should be less than 50%)

sum(large.resid)/nrow(residuals)

**Mean residual value** (should be less than .08?)

sqrt(mean(residuals^2))

hist(residuals) for visual examination

**factor rotation: orthogonal (varimax)**

To carry out a varimax rotation, change the *rotate* option in the *principal()* function from *“none”* to *“varimax”* (you could also exclude it altogether because varimax is the default if the option is not specified):

pc3 <- principal(raqData, nfactors = 4, rotate = "varimax")

pc3

Interpreting the factor loading matrix is a little complex; we can make it easier by using the *print.psych()* function.

Generally you should be very careful with the cut-off value – if you think that a loading of .4 will be interesting, you should use a lower cut-off (say, .3), because you don’t want to miss a loading that was .39:

print.psych(pc3, cut = 0.3, sort = TRUE)

**factor rotation: oblique (oblimin)**

The command for an oblique rotation is very similar to that for an orthogonal rotation – we just change the *rotate* option from *“varimax”* to *“oblimin”*.

pc4 <- principal(raqData, nfactors = 4, rotate = "oblimin")

As with the previous model, we can look at the factor loadings from this model in a nice easy-to-digest format by executing:

print.psych(pc4, cut = 0.3, sort = TRUE)

**So, steps in EFA in R:**

1. Preliminary analyses

kmo(raqData)

raqMatrix<-cor(raqData)

cortest.bartlett(raqData)

det(cor(raqData))

1. Extract/Determine number of factors

pc1 <- principal(raqData, nfactors = 23, rotate = "none")

plot(pc1$values, type = "b")

1. Rotate/Interpret factors

pc4 <- principal(raqData, nfactors = 4, rotate = "oblimin")

print.psych(pc4, cut = 0.3, sort = TRUE)

**Summary FA/PCA**

* Factor vs. Principal Components Analyses
* Exploratory vs. Confirmatory
  + For EFA, Kaiser criterion (eigenvalues), scree plot, a priori
* Orthogonal vs. Oblique rotations
  + Varimax vs. Promax
    - Pattern vs. Structure matrices (for Oblique)
  + Interpret pattern matrix (for oblique) and rotated component matrix (for orthogonal

**Reliability: internal consistency**

* **Split-half method**
  + Splits the questionnaire into two random halves, calculates scores and correlates them.
* **Cronbach’s alpha**
  + Splits the questionnaire into all possible halves, calculates the scores, correlates them and averages the correlation for all splits (well, sort of …).
  + Ranges from 0 (no reliability) to 1 (complete reliability)
  + Reliable is α >.7, but with increasing nr of questions you need a larger α.
  + Remember to treat subscales separately, and reverse score reverse phrased items!! If not, α can be negative.

Creating subsets for the subscales, e.g.:

computerFear<-raqData[, c(6, 7, 10, 13, 14, 15, 18)]

statisticsFear <- raqData[, c(1, 3, 4, 5, 12, 16, 20, 21)]

mathFear <- raqData[, c(8, 11, 17)]

peerEvaluation <- raqData[, c(2, 9, 19, 22, 23)]

**to get the chronbach’s alpha in R:**

The *alpha()* function (in *psych* package) requires the name of the dataframe for each subscale, and, where necessary, to include the *keys* option:

alpha(computerFear)

alpha(statisticsFear, keys = c(1, -1, 1, 1, 1, 1, 1, 1))

alpha(mathFear)

alpha(peerEvaluation)

Note that ggplot2 also has an *alpha* function, so you may have to remind R

psych::alpha()

alternative packages you can use: omega, lambda4.

**Summary of internal reliability**

Cronbach’s alpha might be the only coefficient you will ever need, but…

* + Alternative estimates available for specific situations (Guttman’s split-half, MacDonald’s omega)
  + Some alternatives actually outperform alpha