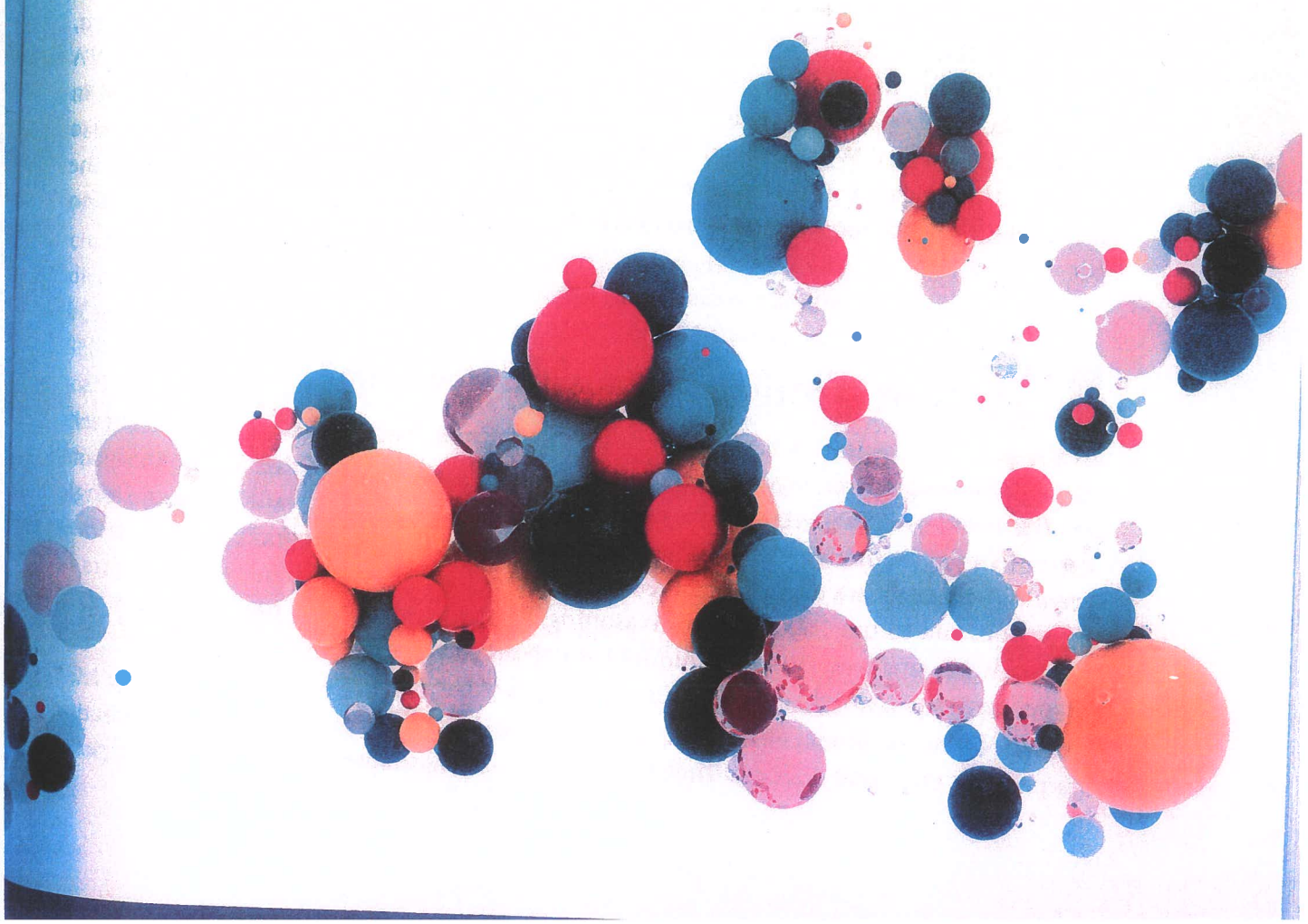
14

Structural Equation Modelling



In this chapter, we make use of the following packages:

- `astatur` - contains the datasets and functions used in this chapter
- `lavaan` - contains functions for estimating CFA/structural equation models

These packages must be installed and activated to run the code provided in this chapter. We can install these by typing the following commands:

```
packages <- c("lavaan", "devtools")
install.packages(packages)
devtools::install_github("ihrke/astatur")
```

Learning outcomes

- Understand the scope of structural equation modelling
- Explain structural equation modelling through confirmatory factor analysis
- Learn to specify, identify, and estimate a structural equation model
- Learn to assess measurement and structural parts of a structural equation model
- Understand and interpret structural equation models using R

In this chapter, we first define what structural equation modelling (SEM) is. We then present several types of SEM, including CFA. Since the CFA model is a very commonly used type of structural equation model, we explain the issues of model specification, model identification, model estimation, model fit measures, and model modification with the help of simple CFA models. In doing so, we also explicate how CFA compares with the EFA that we discussed in the previous chapter. We then go through the SEM process using a latent path model (also called full/complete SEM or structural model) using R. In this review, we focus more on the interpretation of the model parameters rather than on the technical details of the estimation process.

14.1 What Is Structural Equation Modelling?

In the previous chapters, we have presented some of the more traditional statistical techniques (linear regression, logistic regression, multilevel regression, etc.) that are used to examine the relationship between one or more independent (exogenous) variables and a single dependent (endogenous) variable. The independent and dependent variables in the above-mentioned models are directly observed variables such as income, height, weight, years of education, and so on. Following this reasoning, we can refer to these traditional statistical approaches as single-equation techniques with observed variables both on the left-hand side (dependent) and the right-hand side (independent) of the equation.

Like any of the traditional techniques, SEM too can be used for explanation and/or prediction purposes in the social sciences. The difference and a major advantage of SEM (as opposed

to single-equation techniques) is that it allows us to specify and estimate the relationship between a number of independent variables and more than one dependent variable at the same time. Furthermore, while traditional techniques such as regression analysis let one only use observed variables, SEM includes latent, unobserved independent, and dependent variables. As such, in a strict sense, we can refer to SEM as a simultaneous multiple-equation technique including latent variables on both sides of the equations, as graphically portrayed in Figure 14.1. SEM is also referred to as latent variable modelling, covariance structure analysis, and linear structural relationships (LISREL). In a broader sense, SEM as a framework allows one to model observed and latent variables as independent and/or dependent variables. However, we will confine ourselves to the strict definition of SEM in this chapter.

KSI (ξ) = exogenous (latent independent) variable

ETA (η) = endogenous (latent dependent) variable

x = indicator of exogenous variable

y = indicator of endogenous variable

DELTA (δ) = measurement error for x indicator

EPSILON (ε) = measurement error for y indicator

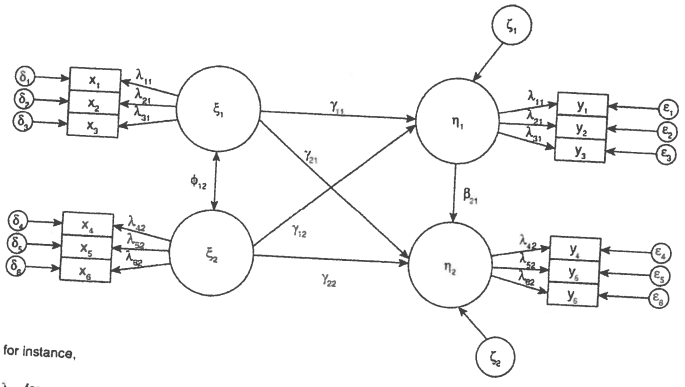
PHI (ϕ) = correlation between exogenous variables

GAMMA (γ) = coefficient between an exogenous and an endogenous variable

BETA (β) = coefficient between two endogenous variables

ZETA (ζ) = unexplained variance in an endogenous variable

LAMBDA (λ) = coefficient (loading) between indicators and latent variables



for instance,

λ_{42}, for exogenous variable, shows the loading of x_4 on the second exogenous variable

λ_{21}, for exogenous variable, shows the loading of y_2 on the first endogenous variable

ϕ_{12} represents the correlation between the first and second exogenous variable

γ_{21} shows the effect of the first exogenous variable on the second endogenous variable

γ_{12} shows the effect of the second exogenous variable on the first endogenous variable

β_{21} shows the effect of the first endogenous variable on the second endogenous variable

Figure 14.1 A structural equation model with LISREL (linear structural relationships) notation

As we can see in Figure 14.1, in the SEM framework, latent variables are represented by large circles, while observed variables are shown by rectangles. One-way arrows (→) represent direct effects, while two-way arrows (↔) represent covariances/correlations. Errors are symbolized by small circles. The notation depicted in Figure 14.1 is known as LISREL. LISREL is an SEM program developed by Jöreskog and Sörbom (1989) and contains a notation that is typically used to graphically portray and mathematically specify all types of structural equation models. Since most of the literature uses the LISREL notation, it is useful to get accustomed to it. Figure 14.1 (left-hand side) also shows naming conventions for the different types of variables and coefficients available in SEM. We will explain all of these terms in more detail later in the chapter, but it is a good idea to keep this figure handy for future reference.

14.1.1 Types of structural equation modelling

SEM can be used to estimate many different kinds of models, including those that contain latent variables. This flexibility is due to advances in specialized computer software. Most models estimated using SEM will fall into one of the following categories:

1. *Confirmatory factor analysis* (CFA) is used to assess a hypothesized latent factor structure containing a set of indicators and one or more latent variables. For instance, we could use CFA to enquire whether the well-known 'Big Five' personality trait factor structure (extraversion, agreeableness, conscientiousness, neuroticism, and openness to experience) would emerge in a particular dataset.
2. *Latent path analysis* (LPA), also referred to as structural regression modelling, full/complete SEM, or combined SEM in the literature, is used not only to examine a factor structure but also to test hypothesized structural relationships among the latent variables. For instance, we could examine whether customer satisfaction is a two-dimensional construct reflecting 'happiness with the product' and 'willingness to recommend the product' as well as assessing whether the former dimension influences the latter.
3. *Latent mean analysis* (LMA) is used to statistically test mean differences between two (or more) groups on latent variables. LMA, like LPA, would also include a CFA as a part of the analysis. We could use LMA to test whether men and women differ in terms of their mean scores on a latent variable such as the 'extraversion' personality trait. LMA can be seen as a direct latent equivalent of traditional ANOVA (R. Kline, 2011). If we further compare men and women in terms of their scores on more than one variable, this will be equivalent to MANOVA (multivariate analysis of variance).
4. *Latent change/growth analysis* is used to test whether there is change in a latent variable over time. It also includes a CFA. For instance, we may try to find out whether a particular organizational intervention (e.g. a novel reward scheme) has been successful in improving employees' job satisfaction (latent variable), which may have been measured at two different time points (for details of such an example see Raykov and Marcoulides, 2006, pp. 5-6).
5. *Latent class analysis* (LCA) is a model-based approach to clustering individuals into groups (i.e. latent classes) based on their responses to a set of observed variables (Wang and Wang, 2012). LCA also includes a CFA. For instance, we may, in an LCA, find out that there are two latent classes (quality-sensitive and price-sensitive customers) emerging from a particular dataset. LCA is not necessarily readily available in all standard SEM software. In R, there are several contributed packages developed specifically for LCA.

The above list can certainly be extended. The first two (CFA and LPA) are, however, the most commonly used SEM techniques in the social sciences. Furthermore, we believe that learning CFA and LPA will lay a sound foundation for understanding the remaining and more advanced SEM techniques. We will, therefore, focus on CFA and LPA in this chapter.

We further suggest that one starts learning SEM through CFA. One reason for this is that it is after all the CFA part of all types of SEM that makes SEM a distinctively special statistical technique as compared with its traditional counterparts (regression, ANOVA, etc.). A second reason is that understanding CFA is an important prerequisite for developing and estimating more complex structural equation models adequately, as problems encountered in SEM stem usually from poorly specified CFA (Bowen and Guo, 2012). A third reason is that all technical issues we will have to tackle for conducting a successful CFA (identification, estimation, etc.) are basically the same for CFA and other SEM techniques. Finally, a standard CFA is a relatively simple case of SEM which may help us to understand some of the complex issues more easily.

14.2 Confirmatory Factor Analysis

CFA is an alternative to or an extension of the EFA that we treated in Chapter 13. Both CFA and EFA belong to the so-called common factor model family, which partitions the variance of an indicator into common/shared variance and unique variance including measurement error (Brown, 2015). In other words, in CFA, the measurement error (unreliability) of indicators is removed during the model estimation. This specific feature of CFA (embedded in SEM) contributes to making structural equation model estimates less biased compared with the traditional techniques such as regression which assume no measurement error at all (Harlow, 2014). This is the main reason for the increasing popularity and application of SEM techniques in social science research.

If EFA removes the measurement error just as CFA does, why do we need CFA? The answer to this question is that CFA is a confirmatory statistical technique which a priori imposes restrictions on the factor model to be estimated (Brown, 2015), while EFA is an inherently exploratory technique. In CFA, we specify the number of factors and pattern of indicator factor loadings beforehand, as well as other model parameters such as those bearing on the independence or covariance of the factors and indicator error variances (Brown, 2015). Specification is only the first step of the entire CFA/SEM process, and it is followed by model identification, model estimation, model assessment, and model modification.

Let us now explain these five steps using a real-life data example. The name of the dataset that we use is `values`, and it is included in the `astatur` package accompanying this book. The data that we use here have been obtained from a survey of 1004 Norwegian individuals. In this survey, the respondents were asked to indicate on an ordinal scale from 1 = *not at all important* to 5 = *very important* how important each of the following personal values was as a guiding principle in their lives: being well respected by others (x_1), a sense of security (x_2), a sense of accomplishment (x_3), self-fulfilment (x_4), and self-respect (x_5). Here, we treat ordinal data as if they were continuous and thus fit a standard linear model. However, if we wanted to fit a model for ordinal data, we would have to use a different estimator (`WLSWM`) than the default `ML` within the `lavaan` package in R.

14.2.1 Model specification

Based on relevant theory, we specify a two-dimensional factor structure for our dataset `values`. This factor structure includes two personal value types (factors): collectivistic values (which x_1 and x_2 load on) and individualistic values (which x_3, x_4, and x_5 load on). We further assume a covariance/correlation between these two factors. No other specification is chosen for our model, which is graphically depicted in Figure 14.2.

Using the LISREL notation presented earlier (see Figure 14.1), we can readily transform our graphically depicted model (Figure 14.2) into regression equations as follows:

$$x_1 = \lambda_{11}\xi_1 + \delta_1. \quad (14.1)$$
$$x_2 = \lambda_{21}\xi_1 + \delta_2. \quad (14.2)$$
$$x_3 = \lambda_{32}\xi_2 + \delta_3. \quad (14.3)$$
$$x_4 = \lambda_{42}\xi_2 + \delta_4. \quad (14.4)$$
$$x_5 = \lambda_{52}\xi_2 + \delta_5. \quad (14.5)$$

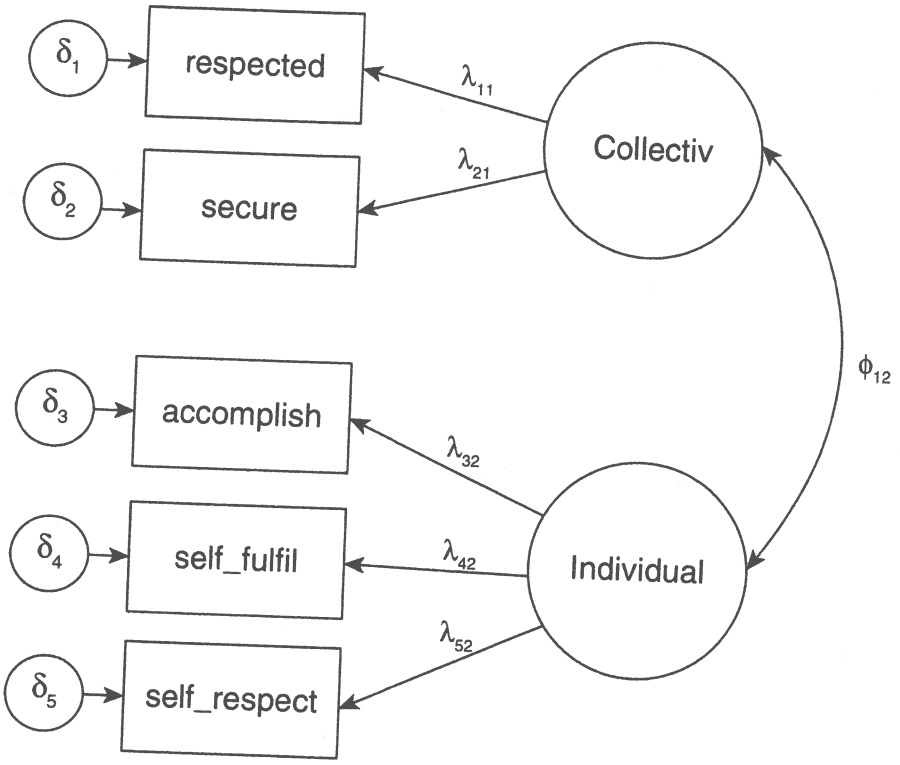


Figure 14.2 Graphical representation of our CFA model for the `values` dataset

Thus, there are five regression models estimated together while also taking into account the correlation between the two factors. This is the reason why we referred to SEM earlier as a simultaneous multiple-equation technique. Listing these equations helps us to understand how SEM works computationally; however, in practice, SEM software does the computation using a compact matrix expression which encompasses all the regression equations at once and ensures fast and efficient calculations. For instance, our model with the five regression models can be presented in one single matrix equation as follows:

$$x = \lambda_x \xi + \delta. \tag{14.6}$$

This matrix equation for our model states that the vector of values for a variable x with components $x_1, \ldots, x_6$ in our raw dataset is a product of the variable's factor loading matrix (λ) on the latent variable vector (ξ) and the vector of scores for cases on that latent variable, plus a vector of error terms (δ; Bowen and Guo, 2012) as follows:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \lambda_{11} & 0 \\ \lambda_{21} & 0 \\ 0 & \lambda_{32} \\ 0 & \lambda_{42} \\ 0 & \lambda_{52} \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} + \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \\ \delta_5 \end{bmatrix}. \tag{14.7}$$

In Equation (14.7), the zeros indicate that x_1 and x_2 do not load on factor 2 (individualistic values) and x_3, x_4, and x_5 do not load on factor 1 (collectivist values). The matrix representation

provides a compact way of specifying quite complex models and can be helpful for both understanding and computations. However, in the remainder of this chapter, we will avoid overly technical discussions of such details and focus on an understanding on a more conceptual level.

14.2.2 Model identification

To allow parameter estimation and model testing in CFA/SEM, the number of freely estimated parameters (unknowns) must not exceed the number of elements (knowns) in the sample variance–covariance matrix denoted **S** (Brown, 2015). The difference between the number of knowns (k) and the number of unknowns (u) is equal to the number of degrees of freedom of the model ($df = k - u$). When $df < 0$, the model is said to be under-identified, when $df = 0$, the model is considered just-identified, and finally, when $df > 0$ the model is referred to as over-identified. Due to the fact that we cannot estimate model parameters for under-identified models and that we cannot test the fit of just-identified models because the fit would be perfect by default (see Raykov and Marcoulides, 2006), we opt for over-identification (R. Kline, 2005).

Take Note!

In this chapter, we use the `lavaan` package (Rosseel, 2012) for estimating CFA and structural equation models. `lavaan` has got two separate functions, `cfa()` for estimating CFA models and `sem()` for estimating latent path or full structural equation models. Although these two functions are currently nearly identical, this may change in the future according to the developer of the package. We will follow this distinction and accordingly use `cfa()` and `sem()` for estimating CFA and full structural equation models, respectively.

Let us now examine our model in Figure 14.2. We can obtain the number of knowns (k) from the formula $(p + 1)/2$, where p denotes the number of indicators (Raykov and Marcoulides, 2006). Since we have five indicators, for our model $k = 5(5 + 1)/2 = 15$. As for the number of unknowns (parameters to estimate), we have three factor loadings (λ_{21}, λ_{42}, λ_{52}), five error variances (δ_1, δ_2, δ_3, δ_4, and δ_5), one covariance (Φ_{12}), and two factor variances (Φ_{11} and Φ_{22}), so that $u = 11$. Then $df = 15 - 1 = 4 > 0$. Although the $df > 0$ rule works fine for CFA/SEM in most instances, in the case of the so-called empirical under-identification, this rule will not be a sufficient criterion to judge the identifiability of a model. Empirical under-identification typically occurs when the covariances in the sample variance–covariance matrix are equal to zero (for further details see Brown, 2015). That being said, we consider our model identifiable, which is necessary for estimating parameters and testing the model fit. R, via the `lavaan` package, provides the df automatically in its estimation output (see Figure 14.3).

In addition to the condition that $df > 0$, latent variables must be assigned a scale/metric for model identification (R. Kline, 2005) since they do not have any metric prior to estimation. There are two main approaches to assigning a metric to a latent variable.

The first approach is to pass on a 'marker' (reference) indicator's metric to the latent variable (Brown, 2015). In R's `lavaan`, the first indicator is by default selected as the marker indicator

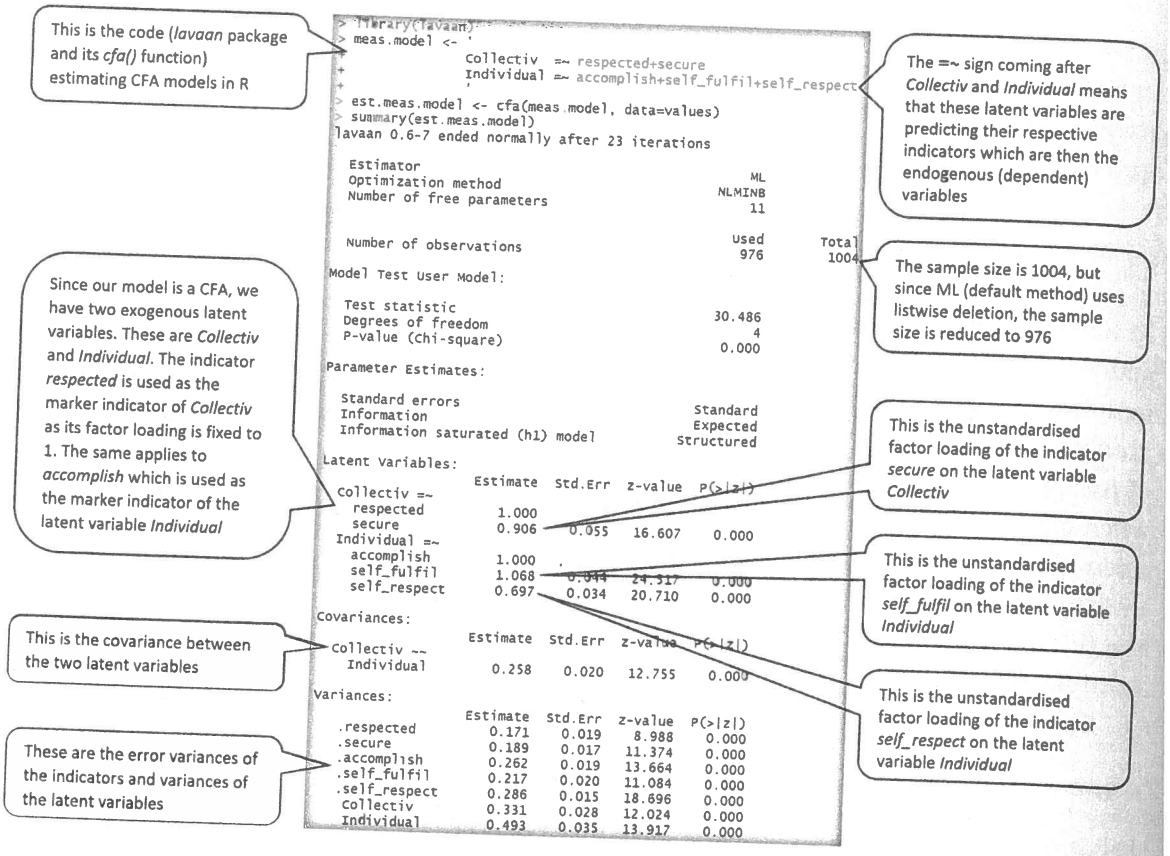


Figure 14.3 R (lavaan) output of a confirmatory factor analysis (CFA) estimated with maximum likelihood (ML; unstandardized solution)

and its unstandardized factor loading is therefore fixed to 1. In our example model, the marker indicators are *respected* and *accomplish* for the latent variables *Collectiv* and *Individual*, respectively (see Figure 14.3). Both of these indicators are measured using an ordinal scale (from 1 to 5) which will also be the scale/metric of the two latent variables. This does not mean that the latent variables will themselves be ordinal. The latent variables will be assumed to have a metric following a normal distribution with mean 0. What we can say about the metric of the latent variables is that the variance of the latent variables will be a portion of the variance of the corresponding marker indicator.

The second approach is that the variance of the latent variable is fixed to 1, meaning that the latent variable is standardized (Brown, 2015), while allowing all the unstandardized factor loadings to be freely estimated. This approach provides semi-standardized coefficients in the solution which are generally of less interest for researchers as they show the change in *Y* (i.e. indicator) in its original units caused by one standard deviation increase in *X* (latent variable). The marker indicator approach, which provides both unstandardized and completely standardized estimates, appears to have been more commonly used in social science publications. This may also be the reason why the marker indicator approach is the default procedure in the `lavaan` package in R.

When fixing a variable to a specific value, there are three types of parameters we can make use of in CFA/SEM. A fixed parameter is a parameter (loading, variance, etc.) that is fixed to a specified value. A free parameter is an unknown element that needs to be estimated. Finally, a constrained parameter is a parameter that is unknown but is constrained to equal one or

more other parameters in the model (Wang and Wang, 2012). The difference between a fixed and a constrained parameter is that the former is not estimated, while the latter is estimated but kept equal for more than one parameter (e.g. factor loadings).

14.2.3 Parameter estimation

The objective of CFA/SEM is to obtain estimates for each parameter of the model (factor loadings, factor variances, etc.) to produce a predicted variance–covariance matrix (denoted by $\boldsymbol{\Sigma}$) that resembles the sample variance–covariance matrix ($\mathbf{S}$) as closely as possible (Brown, 2015, p. 62). In other words, as in ordinary least-squares regression (see Chapter 7), the aim here is also to minimize the difference between the predicted ($\boldsymbol{\Sigma}$) and the observed sample values ($\mathbf{S}$). This minimization is measured using an objective function used for fitting (F). Each estimation method has its own fitting function. The most common estimation method used in CFA/SEM is ML, which applies the fitting function

$$F_{ML} = \ln |\mathbf{S}| - \ln |\boldsymbol{\Sigma}| + \text{trace}(\mathbf{S}\boldsymbol{\Sigma}^{-1}) - p,$$

where $\ln |\mathbf{S}|$ is the natural log of the determinant of $\mathbf{S}$, $\ln |\boldsymbol{\Sigma}|$ is the natural log of the determinant of $\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma}^{-1}$ is the inverse of $\boldsymbol{\Sigma}$, and p is the number of indicators.

When $F_{ML} = 0$, the model fits the data perfectly. However, when the model is over-identified, there is always a degree of mismatch between $\mathbf{S}$ and $\boldsymbol{\Sigma}$. ML uses an iterative procedure to try to find the estimate that minimizes this difference. The smaller the difference, the better the model fits the data.

R (`lavaan`) includes many different estimation methods, a complete list of which can be obtained using `help(lavoptions, package="lavaan")`. In Figure 14.4, we provide an overview of the most common estimation methods (standard and robust versions) depending on type of data (continuous and ordinal/binary) as well as missing data. The default estimation method of the `lavaan` functions (`cfa()` and `sem()`) is ML. However, researchers generally choose `MLR` for continuous and `WLSMV` for ordinal/binary data. Having said that, if the ordinal data have many levels (at least 5) and are approximately normally distributed, analysing a covariance matrix using `ML` does not result in severe bias in fit measures, parameter estimates, or standard errors and can therefore be safely used (Finney and DiStefano, 2013).

We estimate our example model using ML and provide the resulting R output in Figure 14.3.

14.2.4 Model assessment

Model assessment entails the interpretation of parameter estimates and the evaluation of the model fit measures.

Interpreting parameter estimates

It is more common to interpret and report CFA/SEM results based on the standardized solution. Thus, in addition to the unstandardized solution given in Figure 14.3, we can also print the standardized solution for our model by adding the `standardized=TRUE` argument in the `summary()` function after our initial (SEM) estimation. The standardized solution for our current example model is provided in Figure 14.5.

Data \ Estimation	Standard	Robust
Continuous	- Maximum likelihood estimator = "ML" - Generalised least squares estimator = "GLS"	- ML with robust standard errors and a Satorra-Bentler scaled test statistic estimator = "MLM" - ML with robust (Huber-White) standard errors and a scaled test statistic that is (asymptotically) equal to the Yuan-Bentler test statistic estimator = "MLR" - Bootstrap standard errors se = "bootstrap"
Ordinal/Binary	- Diagonally weighted least squares estimator = "DWLS"	- Mean and variance-corrected weighted least squares estimator = "WLSMV"
Missing	- Casewise or full information maximum likelihood (*fiml*) missing = "ML"	

* The lavaan 0.5 series can deal with binary and ordinal (but not nominal) endogenous variables.

Figure 14.4 Overview of estimation methods for structural equation models

Let us now interpret the standardized factor loadings that we can find in the column titled `Std.all` in Figure 14.5. As you can see, none of the factor loadings is fixed to 1, but, instead, they are all freely estimated. The reason is that when we ask for the completely standardized solution, the variance of the latent variable is fixed to 1 as in the case of using the factor variance approach to assign a metric to a latent variable. While in the factor variance approach we standardize only the latent variable, here we standardize both the latent variable and the indicator, giving us the so-called completely standardized estimates.

For instance, the standardized loading of the indicator *respected* on the latent variable *Collectiv* is 0.812. The standardized factor loading of 0.812 can be interpreted as the correlation between the indicator and the latent variable as long as there are no cross-loading indicators in the model. For indicators loading on more than one latent variable, standardized loadings resemble standardized beta coefficients in a multiple regression (Brown, 2015, p. 115). This is the same as saying that one latent variable is predicting the indicator while holding the other latent variable constant (Brown, 2015, p. 131). As such, the squared factor loading of 0.812 is 0.659, indicating that nearly 66% of the variance in the indicator *respected* is explained by the latent variable *Collectiv*. We can interpret the remaining standardized loadings in the same manner. We would generally opt for standardized factor loadings equal to or above 0.4 in CFA/SEM. In Figure 14.5, we observe that all of the standardized loadings are clearly above the threshold of 0.4, lending support to our model.

```
> summary(est.meas.model, standardized=TRUE)
lavaan 0.6-7 ended normally after 23 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of free parameters                         11

  Number of observations                          Used       Total
                                                   976        1004

Model Test User Model:

  Test statistic                                30.486
  Degrees of freedom                                 4
  P-value (Chi-square)                           0.000

Parameter Estimates:

  Standard errors                             Standard
  Information                                 Expected
  Information saturated (h1) model          Structured

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  Collectiv =~
    respected         1.000                               0.576    0.812
    secure            0.906    0.055   16.607    0.000    0.521    0.768
  Individual =~
    accomplish        1.000                               0.702    0.808
    self_fulfil       1.068    0.044   24.517    0.000    0.749    0.849
    self_respect      0.697    0.034   20.710    0.000    0.489    0.675

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  Collectiv ~~
    Individual        0.258    0.020   12.755    0.000    0.638    0.638

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
   .respected         0.171    0.019    8.988    0.000    0.171    0.340
   .secure            0.189    0.017   11.374    0.000    0.189    0.410
   .accomplish        0.262    0.019   13.664    0.000    0.262    0.347
   .self_fulfil       0.217    0.020   11.084    0.000    0.217    0.279
   .self_respect      0.286    0.015   18.696    0.000    0.286    0.545
    Collectiv         0.331    0.028   12.024    0.000    1.000    1.000
    Individual        0.493    0.035   13.917    0.000    1.000    1.000
```

Figure 14.5 R (`lavaan`) output of a confirmatory factor analysis estimated with maximum likelihood (standardized solution)

By the way, the amount of variance in an indicator explained by a latent variable can also be considered as a measure of *indicator reliability* (Brown, 2015). By typing the following command after our SEM estimation in R, you can obtain the complete overview of indicator reliabilities:

```
inspect(est.meas.model, what="rsquare")
##     respected       secure     accomplish     self_fulfil
##         0.660        0.590          0.653          0.721
## self_respect
##         0.455
```

Further down in Figure 14.5 in the column titled `Std.all`, we also observe the amount of variance in each indicator not accounted for by its latent variable. For instance, `.respected`

under *Variances:* shows that 34% of the variance in the indicator *respected* is not explained by the latent variable *Collectiv*. This confirms our interpretation above that 66% of the variance of this indicator is explained.

Having examined the indicator reliabilities, we can further examine factor/scale reliabilities. *Factor/scale* reliability refers to the proportion of the total variation in a scale formed by our indicators that is attributed to the true score (i.e. latent variable; Acock, 2013, p. 20). To examine the reliability of our scales, we will compute and report Raykov's (1997) reliability coefficient (RRC), a measure which is commonly seen as more accurate than Cronbach's alpha. We provide a contributed function called `relicoef()` in the package `astatur` that computes factor reliability coefficients for CFA/SEM factors using Raykov's (1997) formula:

$$\text{RRC} = \frac{(\sum \lambda_i)^2 \phi}{(\sum \lambda_i)^2 \phi + \sum \theta_{ii}}, \tag{14.8}$$

where the λ_i are the unstandardized loadings, ϕ is the factor variance, and θ_{ii} are the unstandardized error variances. In an extension of this formula for factors with correlated errors (at least one error covariance), the equation reads

$$\text{RRC} = \frac{(\sum \lambda_i)^2 \phi}{(\sum \lambda_i)^2 \phi + \sum \theta_{ii} + 2\theta_i}, \tag{14.9}$$

where θ_i are the unstandardized error covariances.

The reliability coefficients of our two latent variables (called *omega*, ω) are computed below. As we observe from the results from `relicoef(est.meas.model)`, the reliability coefficients of *Collective* and *Individual* are 0.770 and 0.831, respectively, both of which are above 0.7 which should be the minimum level of reliability for a CFA/SEM factor/scale:

```
relicoef(est.meas.model)
##        Latent       RRC
## 1  Collectiv 0.7696239
## 2 Individual 0.8311486
```

In addition to indicator reliability and scale reliability, we should also examine the *construct validity* of the latent variables in CFA/SEM. A latent variable can be claimed to be valid when both convergent and discriminant validity are demonstrated. Convergent validity is the extent to which a set of indicators reflecting the same latent variable are positively correlated. Convergent validity is established when a latent variable has (at least) an average correlation (standardized loading) of 0.7 with its corresponding indicators. Squaring this average correlation (0.7^2) would provide us with the average variance extracted (AVE; here 0.5) by the latent variable, meaning that the latent variable should explain (at least) an average of 50% variance in its associated indicators.

Discriminant validity is about the distinctiveness of latent variables. The higher the correlation between a latent variable and its indicators as compared with its correlation with the other indicators in the model, the more distinct the latent variable is. As we have just seen, the AVE is a function of the correlation between a latent variable and its indicators. Furthermore, the squared correlation between two different latent variables indicates how much variance

the latent variables share with each other's indicators. As such, we should expect each of the latent variables' AVE to be larger than the squared correlation between them to establish discriminant validity (Fornell and Larcker, 1981).

Computing the AVE for each latent variable as well as the squared correlation between latent variables can be a tedious task. There is, however, a contributed function called `condisc()` which we include in the `astatur` package accompanying this book. This function can be run in R right after our SEM estimation, applying it to the estimated model object (see code below). According to these results, we can claim that convergent validity is present in that both AVE values are above the suggested minimum level of 0.5. In addition, discriminant validity can also be claimed to be present since the AVEs (0.625 and 0.610) are clearly larger than the squared correlation (0.407) between the two latent variables:

```
condisc(est.meas.model)
## $Squared_Factor_Correlation
##            Cllctv Indvdl
## Collectiv  1.000
## Individual 0.407  1.000
##
## $Average_Variance_Extracted
## Collectiv Individual
##     0.625      0.610
```

Model fit indices

Model fit is the extent to which our model predicts the sample variance–covariance matrix. The way we can measure the model fit is to compare the model-predicted variance–covariance matrix ($\boldsymbol{\Sigma}$) to the sample variance–covariance matrix ($\mathbf{S}$). The smaller the difference between $\boldsymbol{\Sigma}$ and $\mathbf{S}$, the better our model fits the data. There are many kinds of model fit indices proposed in the literature (see West et al., 2012, pp. 212–213), each of which essentially measures the difference between the two matrices ($\boldsymbol{\Sigma} - \mathbf{S}$) in different ways (Bollen, 1989). In the following, we will treat some of the most commonly reported model fit indices which are also provided by R (`lavaan`): the chi-squared (χ^2) test, the standardized root mean square residual (SRMR), root mean square error of approximation (RMSEA), comparative fit index (CFI), Tucker–Lewis index (TLI).

Chi-squared test

The χ^2 test works quite like the F test used for comparing nested models in multiple regression. Since the aim of our hypothesized CFA/structural equation model (HM) is to reproduce $\mathbf{S}$, one way of assessing our HM's performance is to compare our model's log-likelihood (LL) with that of a model that already reproduces $\mathbf{S}$. One such model is the so-called saturated model (SM), fitting the data perfectly (i.e. $df = 0$). An SM includes only variances and covariances/correlations. The χ^2 test for the comparison of the two models (HM and SM) would assess the following:

H_0: HM fits no worse than SM (i.e. $\boldsymbol{\Sigma} = \mathbf{S}$)
H_1: HM fits worse than SM (i.e. $\boldsymbol{\Sigma} \neq \mathbf{S}$).

Having already estimated our HM, we know that its log-likelihood (LL_{HM}) is −4708.866 (s
Figure 14.6).

```
> summary(est.meas.model, fit.measures=TRUE, estimates=FALSE)
lavaan 0.6-7 ended normally after 23 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of free parameters                         11

  Number of observations                          Used       Total
                                                   976        1004

Model Test User Model:

  Test statistic                                30.486
  Degrees of freedom                                 4
  P-value (Chi-square)                           0.000

Model Test Baseline Model:

  Test statistic                              1886.339
  Degrees of freedom                                10
  P-value                                        0.000

User Model versus Baseline Model:

  Comparative Fit Index (CFI)                    0.986
  Tucker-Lewis Index (TLI)                       0.965

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -4708.866
  Loglikelihood unrestricted model (H1)             NA

  Akaike (AIC)                                9439.732
  Bayesian (BIC)                              9493.450
  Sample-size adjusted Bayesian (BIC)         9458.514

Root Mean Square Error of Approximation:

  RMSEA                                          0.082
  90 Percent confidence interval - lower         0.057
  90 Percent confidence interval - upper         0.111
  P-value RMSEA <= 0.05                          0.021

Standardized Root Mean Square Residual:

  SRMR                                           0.028
```

Figure 14.6 R (lavaan) output from maximum-likelihood estimation with fit measures

We can estimate the SM by using the following code:

```
saturated.mod <- '
  Collectiv  =~ 0*respected+0*secure
  Individual =~ 0*accomplish+0*self_fulfil+0*self_respect
```

see

```
# Covariance:
  respected ~~ secure
  respected ~~ accomplish
  respected ~~ self_fulfil
  respected ~~ self_respect
  secure ~~ accomplish
  secure ~~ self_fulfil
  secure ~~ self_respect
  accomplish ~~ self_fulfil
  accomplish ~~ self_respect
  self_fulfil ~~ self_respect
# Means
  respected ~ 1
  secure ~ 1
  accomplish ~ 1 self_fulfil ~ 1
  self_respect ~ 1
  '

est.saturated.mod <- cfa(saturated.mod, data=values)
summary(est.saturated.mod, fit.measures=TRUE)
```

The code above will yield a log-likelihood $LL_{SM} = -4693.623$. The difference between $LL_{HM} - LL_{SM}$ is −15.243. Multiplying this difference by −2 results in the reported χ^2 value (30.486). In addition, we know that the HM has $df = 4$ and the SM has $df = 0$. We can then get the probability of falsely rejecting our null hypothesis above by typing `pchisq(30.486, df=4, lower.tail=FALSE)` in R, which gives us the p value of 0.000. This means that we reject the null hypothesis that our HM fits no worse than the SM, and we conclude that our HM fits worse than the SM. Typically, we would want a non-significant χ^2 to be able to claim that a CFA/structural equation model fits the data well. These results are identical to those readily provided by R (`lavaan`) shown in Figure 14.6. You just type `summary(est.meas.model, fit.measures=TRUE)` after our SEM estimation to obtain the χ^2 test results as well as the remaining default fit indices supplied by R (`lavaan`).

There is a general consensus in the literature that the χ^2 test is highly sensitive to sample sizes in that the χ^2 statistic tends to always be statistically significant in large samples. This is because even very small differences can become statistically significant in large samples. On the other hand, small the samples may obscure poor fit and yield less precise estimates of the parameters in CFA/SEM (West et al., 2012). Thus, it is generally recommended to also examine the alternative model fit indices that we discuss next.

Standardized root mean square residual

The difference between the predicted and the sample variance–covariance matrix is the residual variance–covariance matrix. In R (`lavaan`), we can ask for this (unstandardized) residual matrix by typing `inspect(est.meas.model, what="resid")` after estimating our structural equation model. The residual matrix indicates how well our hypothesized model is doing in terms of prediction. One omnibus way of quantifying the residual matrix is to take

the average of all its elements (i.e. variances and covariances). This quantity is referred to as the root mean square residual (RMR). However, the RMR is computed based on the covariances (i.e. raw units) of $\boldsymbol{\Sigma}$ and $\mathbf{S}$. As there are often indicators with different units of measurement in CFA/SEM, it will be difficult to interpret a given RMR (R. Kline, 2005), precluding comparisons across datasets (West et al., 2012).

One way of overcoming these drawbacks is to take the average of all the elements in the residual matrix, which is arrived at by subtracting predicted and sample correlation (and not covariance) matrices. The resulting quantity is referred to as the SRMR. It shows the average difference between the correlations in $\boldsymbol{\Sigma}$ and $\mathbf{S}$ (Brown, 2015). This index can be used for comparison purposes in CFA/SEM. The smaller the SRMR, the better the model. The SRMR ranges from 0 (best fit) to 1 (worst fit). SRMR < 0.1 is generally associated with acceptable fit in CFA/SEM (Wang and Wang, 2012). When we look at the SRMR of our example model in Figure 14.6, it certainly seems to provide support for a good fit.

Root mean square error of approximation

As opposed to the so-called absolute fit indices like χ^2 and the SRMR, the root mean square error of approximation (RMSEA) takes the model complexity and sample size into consideration in that it penalizes models with too many parameters to estimate (i.e. with low *df*) and accordingly favours simpler models (i.e. models with higher *df*). More specifically, RMSEA compensates for the effect of model complexity by conveying discrepancy in fit ($\boldsymbol{\Sigma} - \mathbf{S}$) per degree of freedom in the model using the following formula (Brown, 2015):

$$\text{RMSEA} = \sqrt{\frac{d}{df_{\text{HM}}}}, \quad \text{where } d = \frac{\left(\chi^2 - df_{\text{HM}}\right)}{N_{\text{HM}}}. \tag{14.10}$$

Let us apply this formula to our model. We have $d = (30.486 - 4)/976 = 0.027$, so

$$\text{RMSEA} = \sqrt{\frac{0.027}{4}} = 0.082\,. \tag{14.11}$$

As can be seen from this formula, as we decrease the *df* (i.e. include more parameters to estimate in our model), the RMSEA value increases. High RMSEA values will be a sign of poor model fit. RMSEA values ≥ 0.10 indicate poor model fit (Browne and Cudeck, cited in Bowen and Guo, 2012, p. 145). As shown in Figure 14.6, our model is associated with an RMSEA value of 0.082, which incidentally is the same as the value that we calculated in Equation (14.11). Since our RMSEA < 0.10, we can claim that our model fit is acceptable.

Comparative fit index

When performing the χ^2 test, we essentially compare our HM with the SM that has perfect fit. Here, we compare our HM with the baseline model (BM, poorest fit) so as to find out the relative improvement in the fit of our HM over that of the BM (R. Kline, 2011). The default BM in `lavaan` is a model that assumes zero covariances/correlations between the indicators. As we have already estimated our HM, we know that its *df* and χ^2 values are 4 and 30.49, respectively (see Figure 14.3). We can also estimate BM by typing

```
baseline.mod <- '
    Collectiv  =~ 0*respected+0*secure
    Individual =~ 0*accomplish+0*self_fulfil+0*self_respect
  # Means
    respected ~ 1
    secure ~ 1
    accomplish ~ 1
    self_fulfil ~ 1
    self_respect ~ 1
    '

est.baseline.mod <- cfa(baseline.mod, data=values)
summary(est.baseline.mod, fit.measures=TRUE)
```

in R to obtain $df = 10$ and $\chi^2 = 1886.34$. Using the formula below, we get the CFI of our estimation:

$$\text{CFI} = 1 - \frac{(\chi^2 - df_{\text{HM}})}{(\chi^2 - df_{\text{BM}})}. \tag{14.12}$$

Let us apply this formula to our model:

$$\text{CFI} = 1 - \frac{30.49 - 4}{1886.34 - 10} = 0.986. \tag{14.13}$$

Observe that this value (0.986) is the same as the CFI provided by R (`lavaan`) in Figure 14.6. The CFI generally ranges from 0 to 1. CFI values ≥ 0.90 are generally associated with acceptable model fit (Acock, 2013). A CFI value of 0.986 indicates that our model does 98.6% better than the worst-fitting model that assumes no correlations among the indicators (Acock, 2013). As such, based on the CFI value of 0.986, we can claim that our model fit is acceptable (in fact, very good).

Tucker-Lewis index

The TLI is another way of comparing our HM with the BM, defined (Wang and Wang, 2012, p. 19) as

$$\text{TLI} = \frac{(\chi^2/df_{\text{HM}}) - (\chi^2/df_{\text{BM}})}{(\chi^2/df_{\text{BM}}) - 1}. \tag{14.14}$$

Again, applying this formula to our model yields

$$\text{TLI} = \frac{(30.49/4) - (1886.34/10)}{(1\,886.34/10) - 1} = 0.965. \tag{14.15}$$

As (14.14) shows, the TLI imposes a penalty for model complexity as the more parameters to estimate, the smaller the value of df_{HM}, thus the larger is (χ^2/df_{HM}), leading to smaller TLI (Wang and Wang, 2012). Notice that the value computed in Equation (14.15) is identical to the TLI computed by R (`lavaan`) in Figure 14.6. The TLI also generally ranges from 0 to 1. TLI values ≥ 0.90 are generally associated with acceptable model fit (Acock, 2013). Based on our large TLI (0.965), we can conclude that our model fit is a good one.

14.2.5 Model modification

Model modification is about changing the specification of a poorly fitting initial CFA/structural equation model in an exploratory manner. Researchers respecify their initial models with the help of the so-called modification indices (MIs) computed by the software. MIs represent the predicted decrease in model χ^2 if a fixed or constrained parameter is freely estimated (R. Kline, 2005, p. 145). MIs are like a cost–benefit analysis in that the cost of letting one parameter be freely estimated is 1 *df*, and the benefit is the reduction that we get in χ^2.

One way of finding out whether the benefit outweighs the cost is to consider the size of the reduction in χ^2. If this reduction is considerably larger than 3.84, we can claim that the benefit outweighs the cost. The value of 3.84 is the critical χ^2 value for $df = 1$. As such, for each modification index larger than 3.84, we would significantly improve the fit of the model by reducing χ^2 significantly (Acock, 2013, p. 26).

Although the fit of our model is not poor and thus there is no need to try to improve its fit, for pedagogical purposes, we will still use our model as an example here to show how to go about using MIs. In R (`lavaan`), we obtain the MIs for our model by typing the following code after our structural equation model estimation:

```
modindices(est.meas.model, minimum.value=3.84)
##           lhs op          rhs     mi    epc sepc.lv
## 15  Collectiv =~  self_fulfil 21.568 -0.330  -0.190
## 16  Collectiv =~ self_respect 15.798  0.220   0.127
## 24     secure ~~  self_fulfil 10.401 -0.036  -0.036
## 25     secure ~~ self_respect  4.495  0.021   0.021
## 26 accomplish ~~  self_fulfil 15.797  0.127   0.127
## 27 accomplish ~~ self_respect 21.568 -0.081  -0.081
##    sepc.all sepc.nox
## 15   -0.215   -0.215
## 16    0.175    0.175
## 24   -0.177   -0.177
## 25    0.089    0.089
## 26    0.532    0.532
## 27   -0.295   -0.295
```

We see that the MI for the suggested correlation between the errors of the indicators `accomplish` and `self_respect` is 21.568. This means that we would reduce χ^2 by 21.568 for 1 *df*, which is considerably larger than 3.84. We can decide to include this correlation in our model in R by typing the following:

```
meas.model2 <- '
               Collectiv  =~ respected+secure
               Individual =~ accomplish+self_fulfil+self_respect
               accomplish ~~ self_respect
               '
est.meas.model2 <- cfa(meas.model2, data=values)
```

To see the model fit after this modification, we simply ask for the model fit indices (as we did earlier) by typing:

```
summary(est.meas.model2, fit.measures=TRUE, estimates=FALSE)
## lavaan 0.6-7 ended normally after 26 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of free parameters                         12
##
##                                                   Used       Total
##   Number of observations                           976        1004
##
## Model Test User Model:
##
##   Test statistic                                 6.701
##   Degrees of freedom                                 3
##   P-value (Chi-square)                           0.082
##
## Model Test Baseline Model:
##
##   Test statistic                              1886.339
##   Degrees of freedom                                10
##   P-value                                        0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.998
##   Tucker-Lewis Index (TLI)                       0.993
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -4696.973
##   Loglikelihood unrestricted model (H1)      -4693.623
##
##   Akaike (AIC)                                9417.947
##   Bayesian (BIC)                              9476.548
##   Sample-size adjusted Bayesian (BIC)         9438.436
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.036
##   90 Percent confidence interval -lower          0.000
##   90 Percent confidence interval -upper          0.072
##   P-value RMSEA <= 0.05                          0.693
```

```
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                          0.012
```

As we can see, including the suggested correlation has indeed improved all of the model fit indices (e.g. RMSEA goes down from 0.082 to 0.036).

Modifying a model based on the MI should be assisted by theoretical considerations, an approach which has been shown to increase the chances of discovering the true model (R. Kline, 2005). A second suggestion is that modifications should be made one at a time, beginning with the largest, because a single change can affect other parts of the solution (Raykov and Marcoulides, 2006). Finally, as it is highly likely that the modification improvements apply to the particular dataset (Raykov and Marcoulides, 2006), modified models should be replicated with independent samples where possible (Chou and Huh, 2012).

Take Note!

We can get all the main results from a CFA/SEM estimation with `cfa()` or `sem()` by using the `summary()` function with the relevant arguments. We can alternatively get the results bit by bit by using distinctive functions. An example of both of these approaches is provided at the end of this chapter.

14.3 Latent Path Analysis

We have covered the SEM process using CFA in the previous section. In doing so, we have explained the SEM issues (from identification to modification) from a theoretical/conceptual perspective and given applications using R (`lavaan`). Since the SEM issues treated in our discussion of CFA in Section 14.2 apply directly to any kind of SEM in general, there is no need to treat these again in this section. Instead, we present an additional example application of SEM using LPA, which is probably the most commonly applied technique in the social sciences.

LPA is used to examine a factor structure as well as testing hypothesized structural relationships. The factor structure is concerned with the relationships between indicators and latent variables, whereas the structural relationships concern links between latent variables. The former is referred to as the measurement part, while the latter is named the structural part; together they constitute LPA.

We will start by presenting the real-life dataset that we use to build up our LPA model. Our dataset called `workout2` is included in the `astatur` package accompanying this book. It was collected from members of a training/fitness centre in 2014 in a medium-sized city in Norway. The members were asked to indicate how well certain features (x_1 and x_2 in Table 14.1) described them as a person, using an ordinal scale (1 = *very badly* to 6 = *very well*). Using a similar scale (1 = *not at all important* to 6 = *very important*), the members were also asked to indicate how important various factors (y_1, . . ., y_9 in Table 14.1) were for working out.

Table 14.1 Overview of the indicators and latent variables for our model

Indicators	Latent variables
x_1 - attractive face	Attractive
x_2 - sexy	
y_1 - to have a good body	Appearance
y_2 - to improve my appearance	
y_3 - to look more attractive	
y_4 - to develop my muscles	Muscle
y_5 - to get stronger	
y_6 - to increase my endurance	
y_7 - to lose weight	Weight
y_8 - to burn calories	
y_9 - to control my weight	

14.3.1 Specification of the LPA model

Based on relevant evolutionary psychology theories, we propose the following hypotheses:

H_1: The more attractive a person perceives herself/himself, the more the person wants to work out to improve her/his physical appearance (i.e. *Attractive* → *Appearance*)
H_2: The more the person wants to work out to improve her/his physical appearance, the more she/he wants to work out to build up muscles (i.e. *Appearance* → *Muscle*)
H_3: The more the person wants to work out to improve her/his physical appearance, the more she/he wants to work out to lose weight (i.e. *Appearance* → *Weight*)
H_4: The more attractive the person perceives herself/himself, the more this will indirectly influence her/him to want to work out more to build up muscles (i.e. *Attractive* → *Appearance* → *Muscle*)
H_5: The more attractive the person perceives herself/himself, the more this will indirectly influence her/him to want to work out more to lose weight (i.e. *Attractive* → *Appearance* → *Weight*).

It is usual in SEM-based publications and in fact quite useful to put together these hypotheses in a path diagram (as done in Figure 14.7) to ease the understanding of the relationships as well as providing a basis for equation-based formulations of these hypotheses.

Using the LISREL notation presented earlier (see Figure 14.1), we can transform our graphical model (Figure 14.7) into regression equations as shown in Table 14.2. As you can see, we give the equations for both the measurement and structural parts. Our structural model can be represented in a single matrix equation as follows:

$$\eta = \beta\eta + \Gamma\xi + \zeta. \tag{14.16}$$

14.3.2 Measurement part

Estimation of our LPA model proceeds in two steps, in that we first establish a psychometrically sound measurement model that is both valid and reliable. Subsequently, we test the

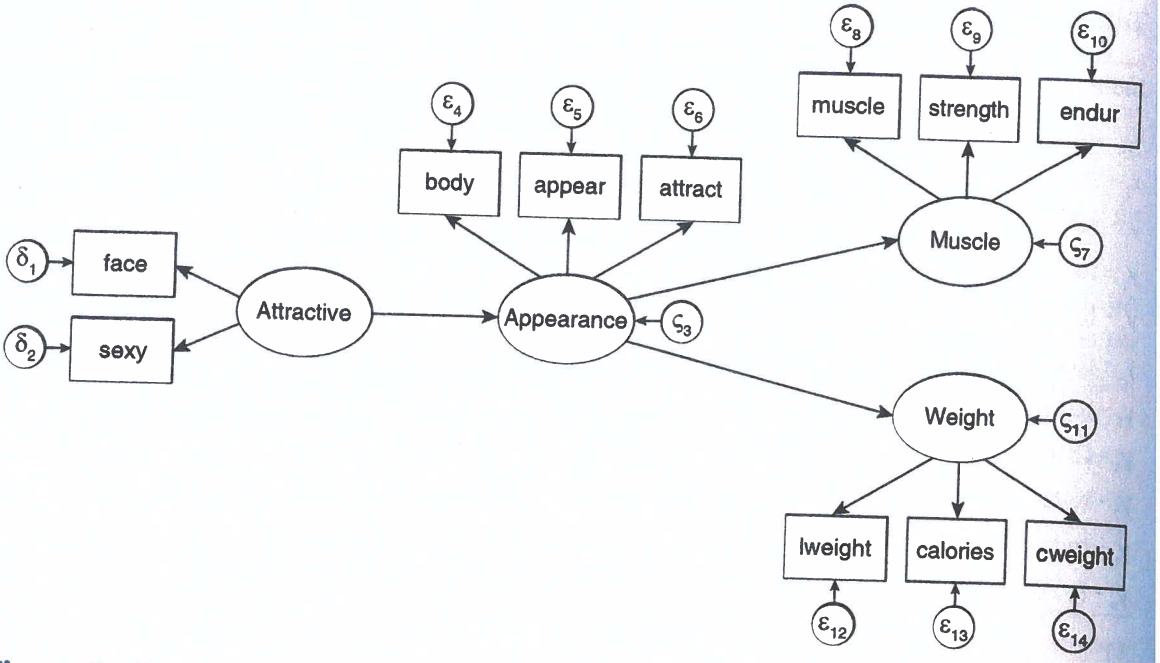


Figure 14.7 Graphical representation of our latent path analysis model

Table 14.2 Equations for the measurement and structural model

Measurement part		Structural part	
Attractive	$x_1 = \lambda_{11}\xi_1 + \delta_1$	Appearance ← Attractive	$\eta_1 = \gamma_{11}\xi_1 + \zeta_3$
	$x_2 = \lambda_{21}\xi_1 + \delta_2$		
Appearance	$y_1 = \lambda_{11}\eta_1 + \varepsilon_4$	Muscle ← Appearance	$\eta_2 = \beta_{21}\eta_1 + \zeta_7$
	$y_2 = \lambda_{21}\eta_1 + \varepsilon_5$		
	$y_3 = \lambda_{31}\eta_1 + \varepsilon_6$		
Muscle	$y_4 = \lambda_{42}\eta_2 + \varepsilon_8$	Weight ← Appearance	$\eta_3 = \beta_{31}\eta_1 + \zeta_{11}$
	$y_5 = \lambda_{52}\eta_2 + \varepsilon_9$		
	$y_6 = \lambda_{62}\eta_2 + \varepsilon_{10}$		
Weight	$y_7 = \lambda_{73}\eta_3 + \varepsilon_{12}$		
	$y_8 = \lambda_{83}\eta_3 + \varepsilon_{13}$		
	$y_9 = \lambda_{93}\eta_3 + \varepsilon_{14}$		

structural model (see Anderson and Gerbing, 1988). The measurement part of the LPA model includes the relationship between each of our four latent variables (*Attractive*, *Appearance*, *Muscle*, and *Weight*) and its respective indicators shown in Figure 14.7. As such, we estimate the measurement model (which essentially is a standard CFA) using the `cfa()` function and then ask for its fit indices using the code below in R:

```
meas.lpa.mod <- '
                Attractive =~ face + sexy
                Appearance =~ body + appear + attract
```

```
                Muscle =~ muscle + strength + endur
                Weight =~ lweight + calories + cweight
                '
est.meas.lpa.mod <- cfa(meas.lpa.mod, data=workout2)
summary(est.meas.lpa.mod, fit.measures=TRUE, standardized=TRUE)
```

We do not show the results (factor loadings, error variances, etc.) obtained from the estimation of the measurement model here. Instead, we present only the model fit indices since our purpose is first to obtain a good fit for the measurement model (Bowen and Guo, 2012, p. 127) prior to examining its psychometric properties. As can be seen in Figure 14.8A, several of our model fit indices (RMSEA > 0.1, TLI < 0.9, etc.) are not acceptable. In order to improve the fit of the measurement model, we ask for the MIs (by typing `modindices(est.meas.lpa.mod, minimum.value=3.84)`) and find that correlating the error variances of two different pairs of indicators (`muscle` and `endur`, `lweigh`, and `body`) would improve the model fit. We estimate the model by adding the first correlation between the error variance of `muscle` and `endur`, and then we estimate the extended model by adding the second correlation between the error variances of `lweigh` and `body`. Making this modification to the measurement model, we re-estimate our respecified/modified model using the following code:

```
meas.lpa.mod2 <- '
                Attractive =~ face + sexy
                Appearance =~ body + appear + attract
                Muscle =~ muscle + strength + endur
                Weight =~ lweight + calories + cweight
                muscle ~~ endur
                lweight ~~ body
                '
est.meas.lpa.mod2 <- cfa(meas.lpa.mod2, data=workout2)
summary(est.meas.lpa.mod2, fit.measures=TRUE, standardized=TRUE)
```

As we can see in Figure 14.8B, the fit indices of our modified model are improved to acceptable levels. For this illustrative example, we accept the RMSEA value of 0.107 in the modified model. However, in reality, we would want RMSEA to be lower than 0.10.

Since we have established an acceptable fit for our modified measurement model, we can now go on to examine its psychometric properties (validity and reliability). Thus, we now present the estimation results of our modified model (i.e. Figure 14.8B). To save space, we show only those parts of the results (i.e. un/standardized loadings) which are of most interest to us here. As can be seen in Figure 14.9, all of the standardized loadings are above the minimum acceptable level of 0.4 and they are all statistically significant.

What is even more important here is to check the convergent and discriminant validity of this modified model. This is easily done in R with the `condisc()` function from the `astatur` package, which yields the output below. This shows that our model exhibits both convergent and discriminant validity. As far as convergent validity is concerned, all of the AVEs are above the suggested level of 0.5. When it comes to discriminant validity, all of the AVEs are considerably larger than all of the squared correlations among the latent variables.

A) Before modification

```
Model Test Baseline Model:

  Test statistic                                1308.679
  Degrees of freedom                              55
  P-value                                          0.000

User Model versus Baseline Model:

  Comparative Fit Index (CFI)                      0.906
  Tucker-Lewis Index (TLI)                         0.864

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)                -3037.463
  Loglikelihood unrestricted model (H1)            NA

  Akaike (AIC)                                  6130.926
  Bayesian (BIC)                                6221.397
  Sample-size adjusted Bayesian (BIC)           6132.709

Root Mean Square Error of Approximation:

  RMSEA                                            0.129
  90 Percent confidence interval - lower           0.108
  90 Percent confidence interval - upper           0.150
  P-value RMSEA <= 0.05                            0.000

Standardized Root Mean Square Residual:

  SRMR                                             0.082
```

B) After modification

```
Model Test Baseline Model:

  Test statistic                                1308.679
  Degrees of freedom                              55
  P-value                                          0.000

User Model versus Baseline Model:

  Comparative Fit Index (CFI)                      0.938
  Tucker-Lewis Index (TLI)                         0.905

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)                -3016.260
  Loglikelihood unrestricted model (H1)            NA

  Akaike (AIC)                                  6092.520
  Bayesian (BIC)                                6189.454
  Sample-size adjusted Bayesian (BIC)           6094.431

Root Mean Square Error of Approximation:

  RMSEA                                            0.107
  90 Percent confidence interval - lower           0.085
  90 Percent confidence interval - upper           0.130
  P-value RMSEA <= 0.05                            0.000

Standardized Root Mean Square Residual:

  SRMR                                             0.074
```

Figure 14.8 Fit indices of the measurement part of latent path analysis model

```
> summary(est.meas.lpa.mod2, standardized=TRUE)
lavaan 0.6-7 ended normally after 49 iterations

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of free parameters                         30

                                                  Used       Total
  Number of observations                           187         246

Model Test User Model:

  Test statistic                               113.576
  Degrees of freedom                            36
  P-value (Chi-square)                           0.000

Parameter Estimates:

  Standard errors                           Standard
  Information                               Expected
  Information saturated (h1) model      Structured

Latent Variables:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  Attractive =~
    face              1.000                               0.710    0.724
    sexy              1.418    0.376    3.769    0.000     1.007    0.922
  Appearance =~
    body              1.000                               1.222    0.814
    appear            1.322    0.078   16.924    0.000     1.615    0.964
    attract           1.252    0.081   15.505    0.000     1.530    0.892
  Muscle =~
    muscle            1.000                               1.505    0.955
    strength          0.530    0.093    5.673    0.000     0.798    0.673
    endur             0.510    0.076    6.737    0.000     0.768    0.728
  Weight =~
    lweight           1.000                               1.489    0.851
    calories          0.995    0.060   16.463    0.000     1.482    0.911
    cweight           0.977    0.062   15.875    0.000     1.454    0.885
```

Figure 14.9 Results of the modified measurement model

```
condisc(est.meas.lpa.mod2)
## $Squared_Factor_Correlation
##              Attrct Apprnc Muscle Weight
## Attractive   1.000
## Appearance   0.063  1.000
## Muscle       0.013  0.208  1.000
## Weight       0.001  0.213  0.069  1.000
##
## $Average_Variance_Extracted
## Attractive Appearance Muscle Weight
##      0.688      0.796  0.631  0.779
```

Finally, we examine the scale reliabilities of the latent variables of our modified measurement model. To compute the reliability coefficients, we just use the `relicoef()` function of the `astatur` package in R, yielding the results below. The results indicate that all of the reliability coefficients are clearly above the recommended threshold of 0.7.

```
relicoef(est.meas.lpa.mod2)
##       Latent       RRC
## 1 Attractive 0.8227873
## 2 Appearance 0.9517422
## 3     Muscle 0.9815475
## 4     Weight 0.9124226
```

14.3.3 Structural part

Given that we have established a sound measurement model, we can now go on to assess the structural part of our model. Thus, we need to estimate the full LPA model. In other words, we extend our modified measurement model with the hypothesized relationships among the latent variables and then estimate the resulting full LPA model displayed in Figure 14.7 (plus the correlations between the error variances) using the following code in R, yielding the output shown in Figure 14.10:

```
full.lpa.mod <- '
              #Measurement model (latent variables)
                Attractive =~ face + sexy
                Appearance =~ body + appear + attract
                Muscle =~ muscle + strength + endur
                Weight =~ lweight + calories + cweight
                muscle ~~ endur
                lweight ~~ body
                Muscle ~~ 0*Weight #set covariance to 0
              #Structural model (regressions)
                Appearance ~ Attractive
                Muscle ~ Appearance
                Weight ~ Appearance
                '

est.full.lpa.mod <- sem(full.lpa.mod, data=workout2)
summary(est.full.lpa.mod, fit.measures=TRUE, standardized=TRUE)
```

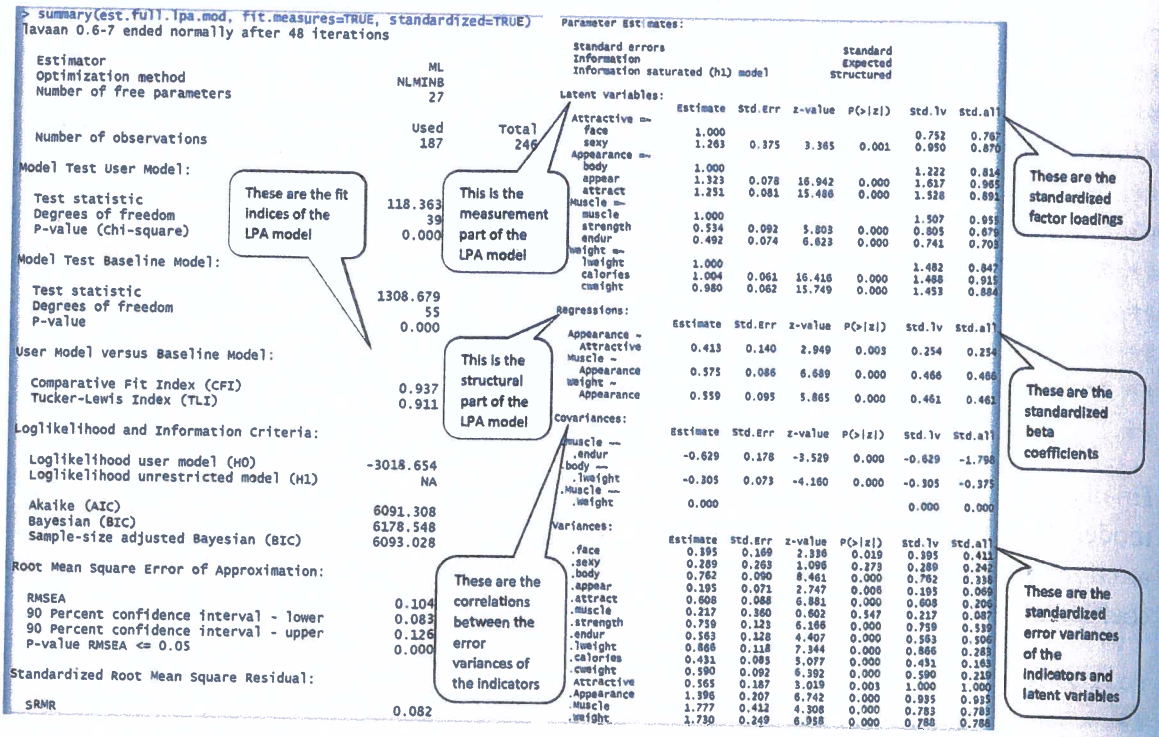


Figure 14.10 The estimation results of the latent path analysis model

The first step is to examine the fit of the LPA model. Due to sample size sensitivity, we do not base our evaluation of the model fit on the χ^2 test. As can be seen in Figure 14.10, RMSEA is just on the edge of being acceptable. Given that the CFI and TLI are both above 0.9 and the SRMR is less than 0.1, we would conclude that the fit of our LPA model is satisfactory, a condition which is generally necessary if we want to go on to examine and interpret the estimates.

The assessment of the structural part is similar to that when examining a statistical model tested using linear regression analysis (see Chapters 7 and 8). First, the 3Ss (sign, significance, and size) of path coefficients should be considered. Path coefficients are estimates that help us to assess the hypothesized relationships in the structural part. These path coefficients are typically presented in a standardized form (as in Figure 14.10), which is equivalent to standardized betas in linear regression. Standardized coefficients range typically between –1 and 1(but are not constrained to that interval). The closer a path coefficient is to ±1, the stronger the relationship (positive/negative) is. And naturally, the closer a path coefficient is to 0, the weaker the relationship is. Standardized beta coefficients equal to or less than 0.09 indicate a small effect, coefficients between 0.1 and 0.2 indicate a moderate effect, and coefficients larger than 0.2 indicate a large effect (see Chapter 8).

Take Note!

The *p* values in the standard `lavaan` output are based on the unstandardized estimates. If you want to get the *p* values for the standardized estimates, you could use the `standardizedsolution(est.full.lpa.mod)` function for this purpose.

Turning to the standardized beta coefficients of our model in Figure 14.10, we observe that all the signs of the coefficients are in the hypothesized direction. That is, *Attractive* has a large and positive effect on *Appearance*, and *Appearance* has a large and positive effect on both *Muscle* and *Weight*. Finally, all of the coefficients are statistically significant at $\alpha = 0.01$. All these findings provide clear support for the first three of our study hypotheses (H_1, H_2, and H_3). Furthermore, we can also ask for the R^2 values for the dependent/endogenous variables of our model. This is done by typing the following command:

```
inspect(est.full.lpa.mod, what="rsquare")
##    face        sexy    body  appear  attract
##   0.589       0.758   0.662   0.931    0.794
##  muscle    strength   endur lweight calories
##   0.913       0.461   0.494   0.717    0.837
## cweight Appearance Muscle  Weight
##   0.781       0.065   0.217   0.212
```

Here, we see that *Attractive* alone explains 6.5% of the variance in *Appearance*, whereas *Appearance* solely explains about 21.7% and 21.2% of the variance in *Muscle* and *Weight*, respectively.

Turning to the last two hypotheses (H_4 and H_5), we need to estimate the indirect effect of *Attractive* (via *Appearance*) on *Muscle* and *Weight*. To do so in R (`lavaan`), we first label all the path coefficients (e.g. *a*, *b*1, and *b*2). We then use these labels to create new parameters (e.g. ind1 and ind2) by using the `:=` operator below, a procedure which yields the results (both unstandardized and standardized) in Figure 14.11. Here, we observe that *Attractive* has a moderate and positive indirect effect on both *Muscle* and *Weight*, and that these indirect effects are statistically significant at the 0.01 level providing support for H_4 and H_5. The standard errors for these newly created parameters are by default estimated using the Delta method. Nevertheless, as with other CFA and structural equation models, bootstrap standard errors can be alternatively estimated by adding the `se = "bootstrap"` argument in the estimation function, `cfa()` or `sem()`.

```
full.lpa.mod2 <- '
                #Measurement model (latent variables)
                  Attractive =~ face + sexy
                  Appearance =~ body + appear + attract
                  Muscle =~ muscle + strength + endur
                  Weight =~ lweight + calories + cweight
                  muscle ~~ endur
                  lweight ~~ body
                  Muscle ~~ 0*Weight #set covariance to 0
                #Structural model (regressions)
                  Appearance ~ a*Attractive
                  Muscle ~ b1*Appearance
                  Weight ~ b2*Appearance
```

```
                #Indirect effects
                  #of Attraction on Muscle
                  ind1 := a*b1
                  #of Attraction on Weight
                  ind2 := a*b2
                  '
est.full.lpa.mod2 <- sem(full.lpa.mod2, data=workout2)
summary(est.full.lpa.mod2, standardized=TRUE)
```

```
Defined Parameters:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
    ab1               0.238    0.086    2.753    0.006    0.119    0.119
    ab2               0.231    0.086    2.702    0.007    0.117    0.117
```

Figure 14.11 The indirect effects (the remaining estimates are omitted)

14.4 Conclusion

We have provided in this chapter a compact introduction to SEM through two of its most commonly applied techniques, CFA and LPA. In a broader sense, however, SEM should be seen as a statistical framework (rather than a single technique) that can replace most of the traditional statistical techniques (regression, ANOVA, logistic regression, etc.) as well as their extensions (seemingly unrelated regression, MANOVA, multinomial logistic regression, etc.). In other words, SEM can be used to estimate any model that includes any number of independent and dependent variables of only observed or of only latent nature as well as a combination of these. This is particularly feasible/testable through R's powerful `lavaan` package.

Key terms

CFA A model examining the relationship between indicators and latent variables
Convergent validity The extent to which a set of indicators reflecting the same latent variable are positively correlated
Discriminant validity The extent to which a latent variable is correlated with its indicators as opposed to the indicators of another latent variable
Endogenous variable An outcome variable
Exogenous variable A predictor variable
Indicator reliability The amount of variance in an indicator explained by a latent variable
Indicator variable A measured variable regressed on the latent variable
Indirect effect The effect of a variable (via another variable) on a dependent variable
Just-identified model Associated with *df* equal to 0
Latent variable An unmeasured variable predicting the indicator variable
Measurement error Represents an unreliable portion of variance of an indicator variable
Measurement model Includes the relationship between latent variables and their indicators
Over-identified model Associated with *df* larger than 0

Scale reliability The proportion of the total variation in a scale formed by our indicators that is attributed to the true score (i.e. latent variable)
Structural model Includes the relationship between latent variables
Under-identified model Associated with *df* less than 0

Questions

1 Explain the criteria for assessing the performance of a CFA as well as a latent path analysis.
2 Try to build and estimate alternative CFA models to the one that we estimate early in this chapter using the same dataset.
3 Explain why structural equation modelling can be used as a substitute for traditional analyses such as the *t* test, ANOVA, and linear regression.
4 Use the `sem()` function to estimate a standard regression model and compare its results with what you obtain by estimating the same model using the `lm()` function.
5 Find and evaluate an article in your field that has applied a CFA or an LPA model.

Further reading

Bowen, N. K. and Guo, S. (2012). *Structural equation modeling*. Oxford University Press, New York, NY.
This book provides a matrix-based explanation of SEM using several applied examples. The book also has a brief chapter on preparation for SEM analyses that nicely supplements this chapter.

Brown, T. A. (2015). *Confirmatory factor analysis for applied research* (2nd ed.). Guilford Press, New York, NY.
Despite the fact that this is a book solely on confirmatory factor analysis, a thorough reading of it will enable you to understand SEM in depth. Working through the examples in the book using R(`lavaan`) would provide useful practice. The book treats some further advanced topics in SEM as well.

Gana, K. and Broc, G. (2019). *Structural equation modeling with lavaan*. Wiley-ISTE, London.
This book provides a comprehensive overview of the features of the `lavaan` package in R. It also presents, through examples, estimation of path models, dyadic models, confirmatory factor analysis, full structural equation models, and more complex models such as latent growth models.

Examples of functions used in this chapter

lavaan

```
meas.model <- '
              Collectiv  =~ respected+secure
              Individual =~ accomplish+self_fulfil+self_respect
              '
est.meas.model <- cfa(meas.model, data=values)
```

- specification and estimation of a CFA model using `cfa()`

```
full.lpa.mod <- '
                #Measurement model (latent variables)
                  Attractive =~ face + sexy
                  Appearance =~ body + appear + attract
                  Muscle =~ muscle + strength + endur
                  Weight =~ lweight + calories + cweight
                  muscle ~~ endur
                  lweight ~~ body
                  Muscle ~~ 0*Weight #set covariance to 0
                #Structural model (regressions)
                  Appearance ~ Attractive
                  Muscle ~ Appearance
                  Weight ~ Appearance
                  '
est.full.lpa.mod <- sem(full.lpa.mod, data=workout2)
```

- specification and estimation of a full structural equation model using `sem()`

```
full.lpa.mod2 <- '
                #Measurement model (latent variables)
                  Attractive =~ face + sexy
                  Appearance =~ body + appear + attract
                  Muscle =~ muscle + strength + endur
                  Weight =~ lweight + calories + cweight
                  muscle ~~ endur
                  lweight ~~ body
                  Muscle ~~ 0*Weight #set covariance to 0
                #Structural model (regressions)
                  Appearance ~ a*Attractive
                  Muscle ~ b1*Appearance
                  Weight ~ b2*Appearance
                #Indirect effects
                  #of Attraction on Muscle
                  ind1 := a*b1
                  #of Attraction on Weight
                  ind2 := a*b2
                  '
est.full.lpa.mod2 <- sem(full.lpa.mod2, data=workout2)
```

- specification and estimation of a mediation model using `sem()`

```
summary(est.full.lpa.mod, standardized=TRUE, ci=TRUE,
        fit.measures=TRUE, modindices=TRUE, rsquare=TRUE)
```

- provides detailed results of a model estimated using `cfa()` or `sem()`

```
parameterestimates(est.full.lpa.mod)
```

- provides CI for the unstandardized estimates

```
standardizedsolution(est.full.lpa.mod)
```

- provides CI for the standardized estimates

```
fitmeasures(est.full.lpa.mod)
```

- provides fit measures

```
modindices(est.full.lpa.mod, sort.=TRUE, minimum.value = 3.84)
```

- provides MIs > 3.84 in descending order

```
inspect(est.full.lpa.mod, what="rsquare")
```

- provides R^2 values

astatur

```
condisc(est.meas.lpa.mod2)
```

- examines convergent and discriminant validity

```
relicoef(est.meas.lpa.mod2)
```

- provides reliability coefficients