Example 2.4.1 - Fitting a path model with observed variables only, extracting model parameter matrices and tables

Fitting a path model with observed variables only

First, we execute the code from the book. We load the package:

```
library("lavaan")
library("knitr")
```

Next, we load the input data (a covariance matrix, in this case):

```
beaujean.cov <- lav_matrix_lower2full(c(648.07, 30.05, 8.64, 140.18, 25.57, 233.21))
colnames(beaujean.cov) <- rownames(beaujean.cov) <- c("salary", "school", "iq")
```

Next, we specify the model:

```
beaujean.model <- '
  salary ~ a*school + c*iq
  iq ~ b*school # this was reversed in first printing of the book
  ind:= b*c
'
```

Finally, we fit the model to the data and inspect the result:

```
beaujean.fit <- sem(beaujean.model, sample.cov=beaujean.cov, sample.nobs=300)
summary(beaujean.fit, standardized = TRUE, fit.measures = TRUE, rsquare = TRUE,
        modindices = TRUE)
```

```
## lavaan 0.6-18 ended normally after 1 iteration
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         5
##
##   Number of observations                           300
##
## Model Test User Model:
##
##   Test statistic                                 0.000
##   Degrees of freedom                                 0
##
## Model Test Baseline Model:
##
```

```
##   Test statistic                                179.791
##   Degrees of freedom                                  3
##   P-value                                         0.000
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                     1.000
##   Tucker-Lewis Index (TLI)                        1.000
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)              -2549.357
##   Loglikelihood unrestricted model (H1)      -2549.357
##
##   Akaike (AIC)                                5108.713
##   Bayesian (BIC)                              5127.232
##   Sample-size adjusted Bayesian (SABIC)       5111.375
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.000
##   90 Percent confidence interval - lower         0.000
##   90 Percent confidence interval - upper         0.000
##   P-value H_0: RMSEA <= 0.050                       NA
##   P-value H_0: RMSEA >= 0.080                       NA
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##   salary ~
##     school     (a)    2.515    0.549    4.585    0.000    2.515    0.290
##     iq         (c)    0.325    0.106    3.081    0.002    0.325    0.195
##   iq ~
##     school     (b)    2.959    0.247   12.005    0.000    2.959    0.570
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##    .salary          525.128   42.877   12.247    0.000  525.128    0.813
##    .iq              157.011   12.820   12.247    0.000  157.011    0.676
##
## R-Square:
##                    Estimate
##     salary            0.187
##     iq                0.324
##
```

```
## Defined Parameters:
##                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
##     ind              0.963    0.323    2.984    0.003    0.963    0.111
##
## Modification Indices:
##
## [1] lhs      op       rhs      mi       epc      sepc.lv  sepc.all sepc.nox
## <0 rows> (or 0-length row.names)
```

Extracting matrices with parameter estimates

The parameter estimates (or coefficients) can be obtained from the fitted model as follows:

```
coefs <- inspect(beaujean.fit, "coef")
coefs$beta
```

```
##        salary    iq school
## salary      0 0.325  2.515
## iq          0 0.000  2.959
## school      0 0.000  0.000
```

```
coefs$psi
```

```
##         salary      iq  school
## salary 525.128
## iq       0.000 157.011
## school   0.000   0.000   8.611
```

Here we see that ψ is a symmetric matrix, giving the (co)variances; β is a non-symmetric matrix, giving the regression coefficients.

Getting a nice-looking table for a paper

To copy the results to a paper, we use function `kable` from package `knitr`:

```
kable(parameterEstimates(beaujean.fit), digits = 3)
```

lhs	op	rhs	label	est	se	z	pvalue	ci.lower	ci.upper
salary	~	school	a	2.515	0.549	4.585	0.000	1.440	3.590
salary	~	iq	c	0.325	0.106	3.081	0.002	0.118	0.532
iq	~	school	b	2.959	0.247	12.005	0.000	2.476	3.443
salary	~~	salary		525.128	42.877	12.247	0.000	441.092	609.165
iq	~~	iq		157.011	12.820	12.247	0.000	131.884	182.137
school	~~	school		8.611	0.000	NA	NA	8.611	8.611
ind	:=	b*c	ind	0.963	0.323	2.984	0.003	0.330	1.595

Note that you can also compile Markdown documents (.Rmd files) as Word documents. As you can see in the github repository, the documents for this example are available as an R Markdown file, which allows you

to combine text, code and results. The code can be run and the document compiled in R Studio by clicking the "Knit" button.

Check out the Markdown file and compare to the compiled .docx and .pdf files to see how the look of the final document can be controlled.

The table still needs some manual adjustments (e.g., p-values should never be written as 0.000, but as $< .001$). Note that you can use function `kable` from package `kableExtra` to have more control over how the final tables look. This can be very helpful for publications (but outside the scope of this course).