semPlot: Unified visualizations of Structural Equation Models

Sacha Epskamp

University of Amsterdam
Department of Psychological Methods

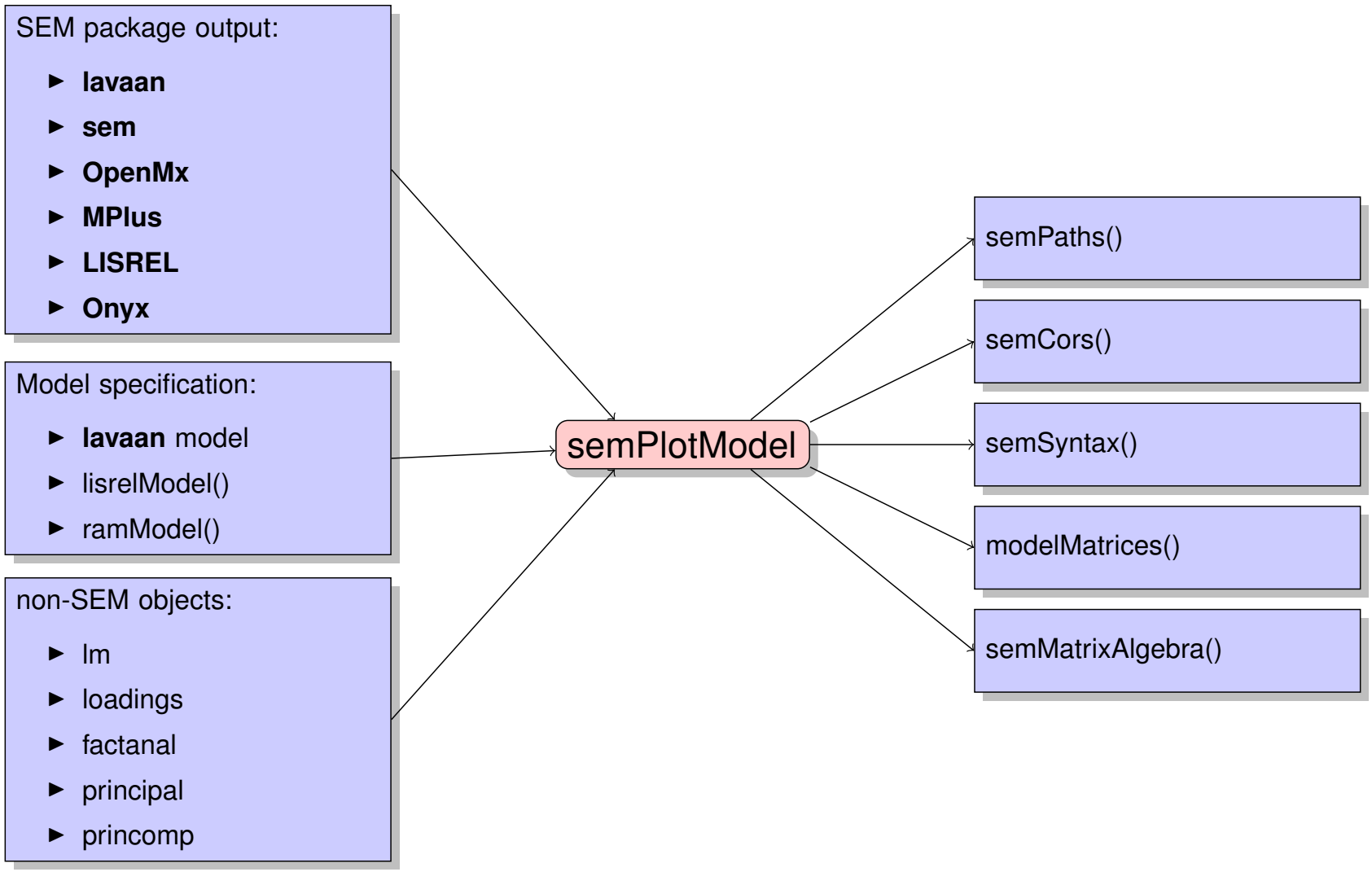
Psychoco 2014
13-02-2014

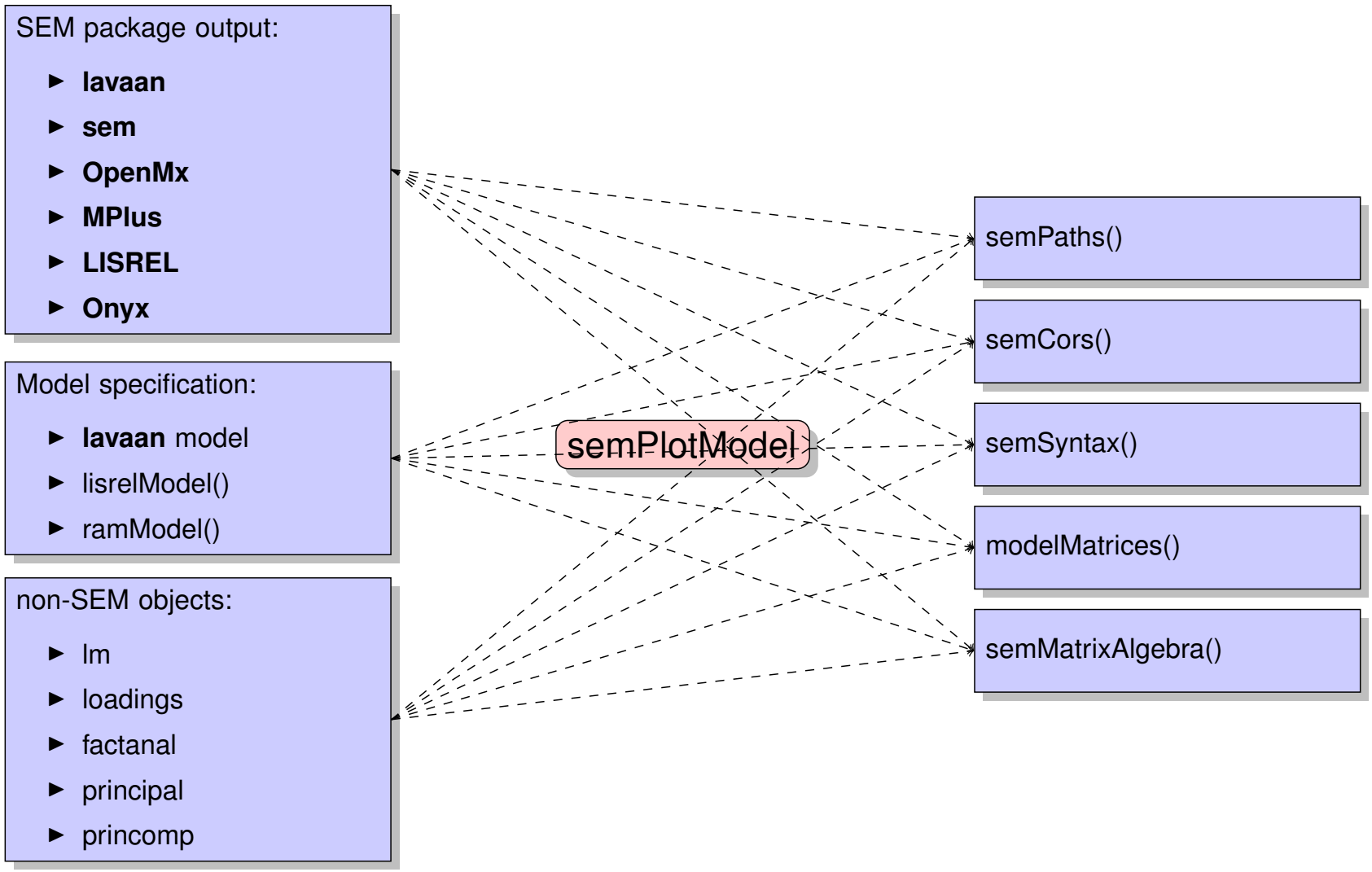
semPlot

- **R** package dedicated to visualizing structural equation models (SEM)
- fills the gap between advanced, but time-consuming, graphical software and the limited graphics produced automatically by SEM software
- Also unifies different SEM software packages and model frameworks in **R**
  - General framework for extracting parameters from different SEM software packages to different SEM modeling frameworks
- Sister package and extension to **qgraph** (Epskamp, Cramer, Waldorp, Schmittmann, & Borsboom, 2012)

Supported input

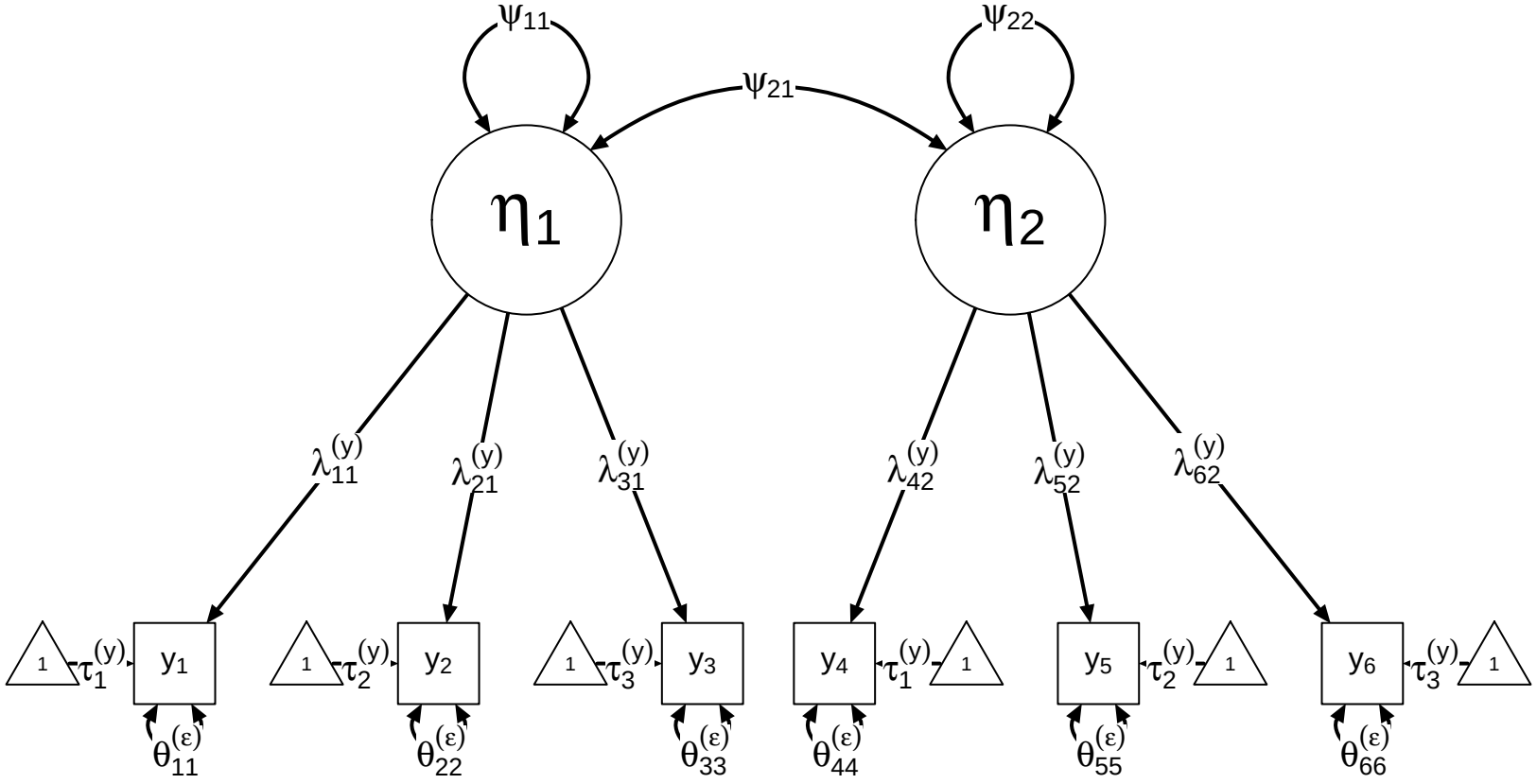
- **R** (R Core Team, 2013) objects:
  - `lm`
  - `loadings`
  - `factanal`
  - `princomp`
  - `principal` (Revelle, 2010)
- **R** package output:
  - **lavaan** (Rosseel, 2012)
    - Output and model
  - **sem** (Fox, Nie, & Byrnes, 2013)
  - **OpenMx** (Boker et al., 2011)
    - Path specification only
- String indication output file of:
  - **MPlus** (L. K. Muthén & B. O. Muthén, 1998–2012)
    - Via **MplusAutomation** (Hallquist & Wiley, 2013)
  - **LISREL** (Jöreskog & Sörbom, 1996)
    - Via **lisrelToR** (Epskamp, 2013)


SEM package output:
lavaan
sem
OpenMx
MPlus
LISREL
Onyx
Model specification:
lavaan model
lisrelModel()
ramModel()
non-SEM objects:
lm
loadings
factanal
principal
princomp
semPlotModel
semPaths()
semCors()
semSyntax()
modelMatrices()
semMatrixAlgebra()


SEM package output:
lavaan
sem
OpenMx
MPlus
LISREL
Onyx
Model specification:
lavaan model
lisrelModel()
ramModel()
non-SEM objects:
lm
loadings
factanal
principal
princomp
semPlotModel
semPaths()
semCors()
semSyntax()
modelMatrices()
semMatrixAlgebra()

Components of a SEM model

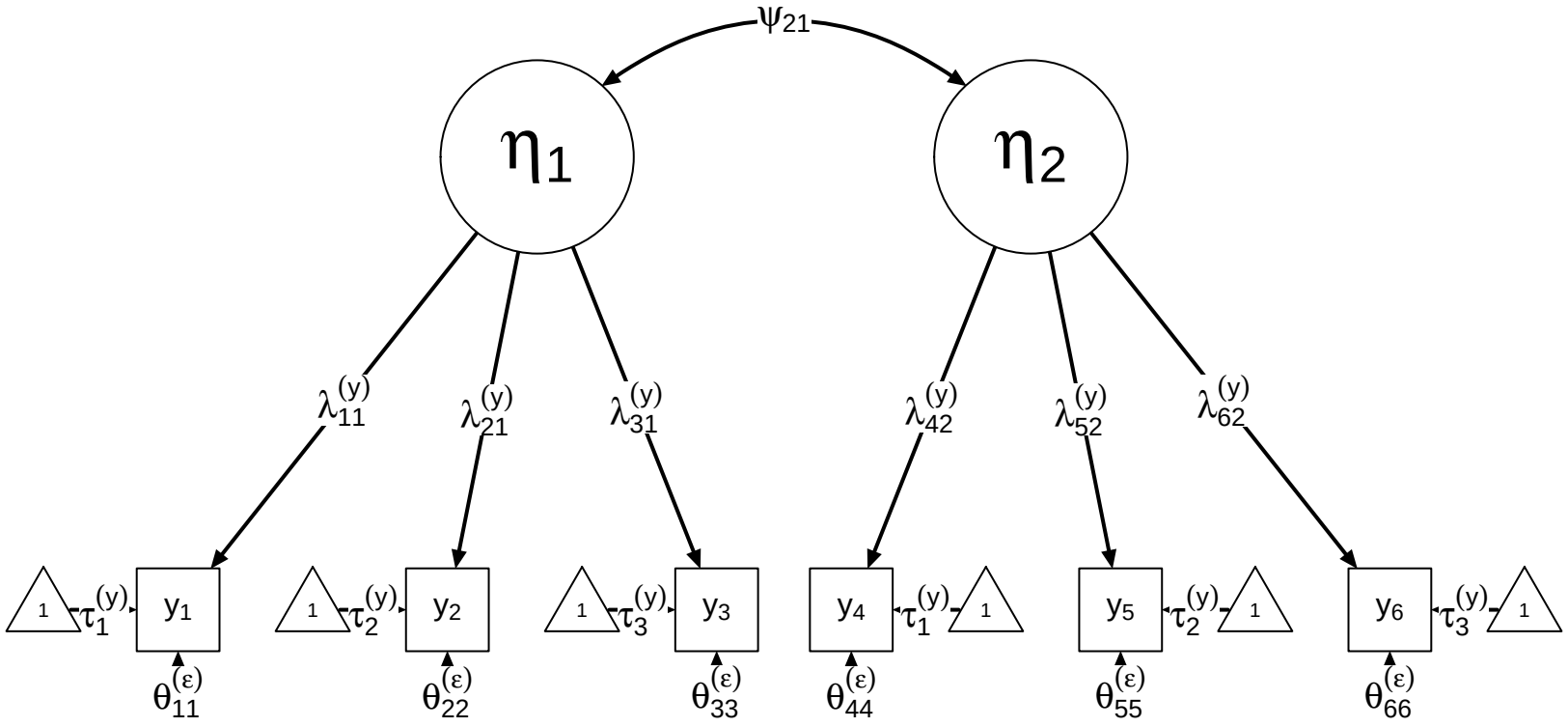
- *Square nodes* indicate manifest or observed variables
- *Circular nodes* indicate latent or unobserved variables
- *Triangular nodes* indicate constant variables (intercepts)
- *Directed edges* indicate linear regression parameters
- *Bidirectional edges* indicate (co)variances
- (Residual) variances can be indicated in several ways:
  - Double headed selfloops (RAM style)
  - Incoming edge with no origin on endogenous variables only (LISREL style)
  - As a latent variable (not yet supported in semPlot)



$$y_1 = \tau_1 + \lambda_{11}\eta_1 + \varepsilon_1$$

$$\vdots$$

$$y_6 = \tau_6 + \lambda_{62}\eta_2 + \varepsilon_6$$



$$y_1 = \tau_1 + \lambda_{11}\eta_1 + \varepsilon_1$$

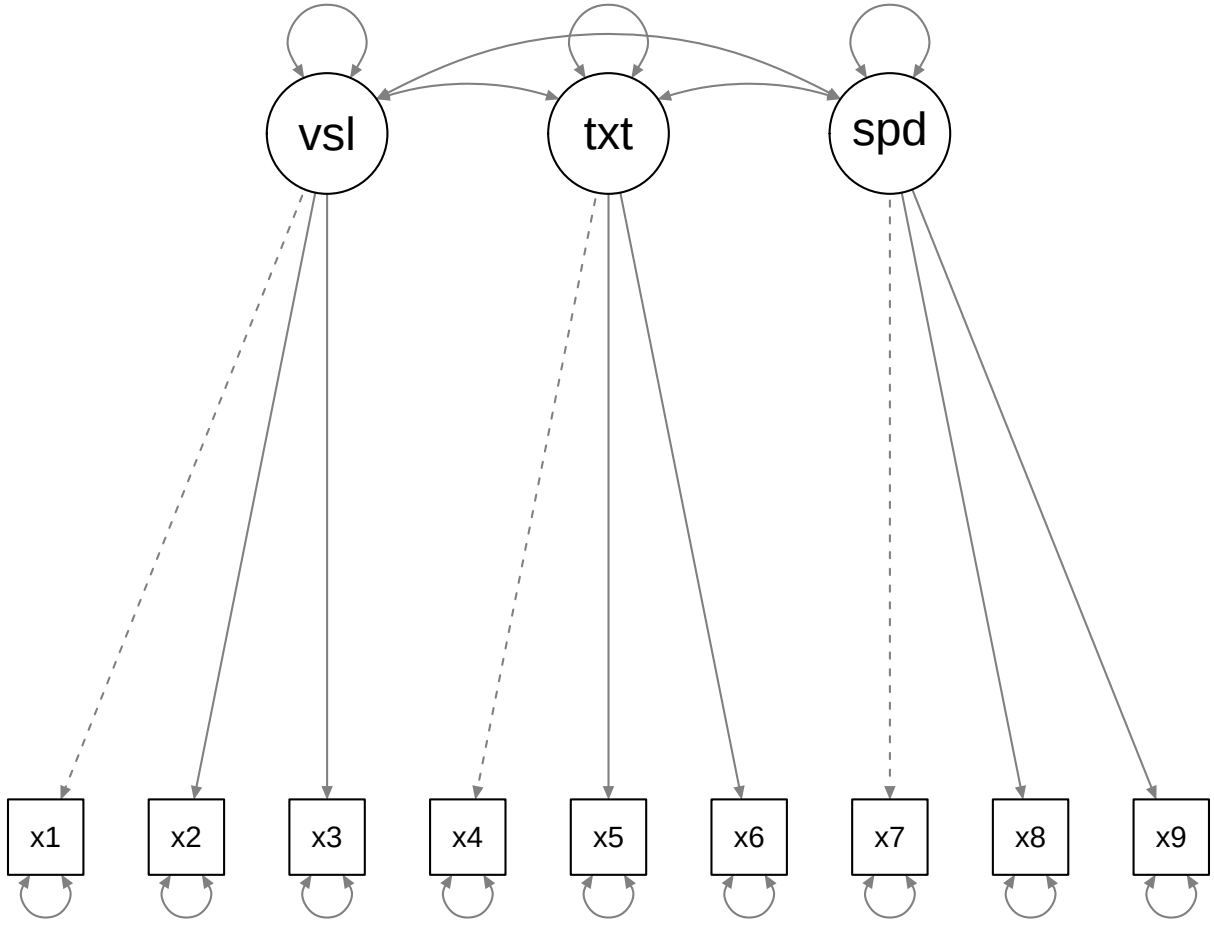
$$\vdots$$

$$y_6 = \tau_6 + \lambda_{62}\eta_2 + \varepsilon_6$$

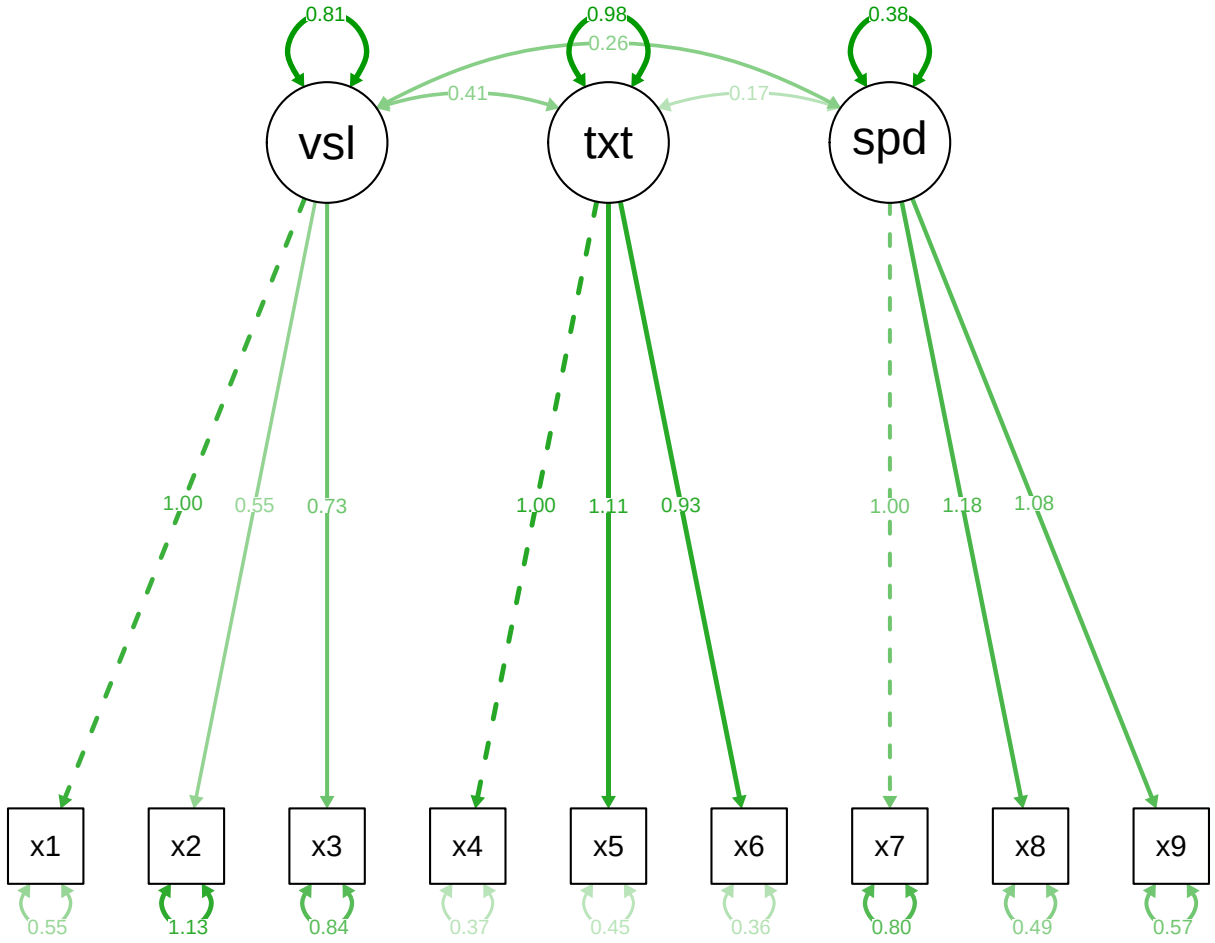
- `semPaths` can be used to to plot path diagrams
- The first argument can be a `semPlotModel` or any input option
- The second argument specifies what the *edge color and width* represent
  - `path`, `diagram` or `mod`
  - `est` or `par`
  - `stand` or `std`
  - `eq` or `cons`
  - `col`
- The third argument specifies what the *edge labels* represent
  - `name`, `label`, `path` or `diagram`
  - `est` or `par`
  - `stand` or `std`
  - `eq` or `cons`
  - `no`, `omit`, `hide` or `invisible`
- These arguments use fuzzy matching
- To visualize parameter estimates I recommend setting edge weights to standardized estimates and edge labels to estimates

```r
library("lavaan")
## The famous Holzinger and Swineford (1939) example
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(HS.model, data=HolzingerSwineford1939)
```

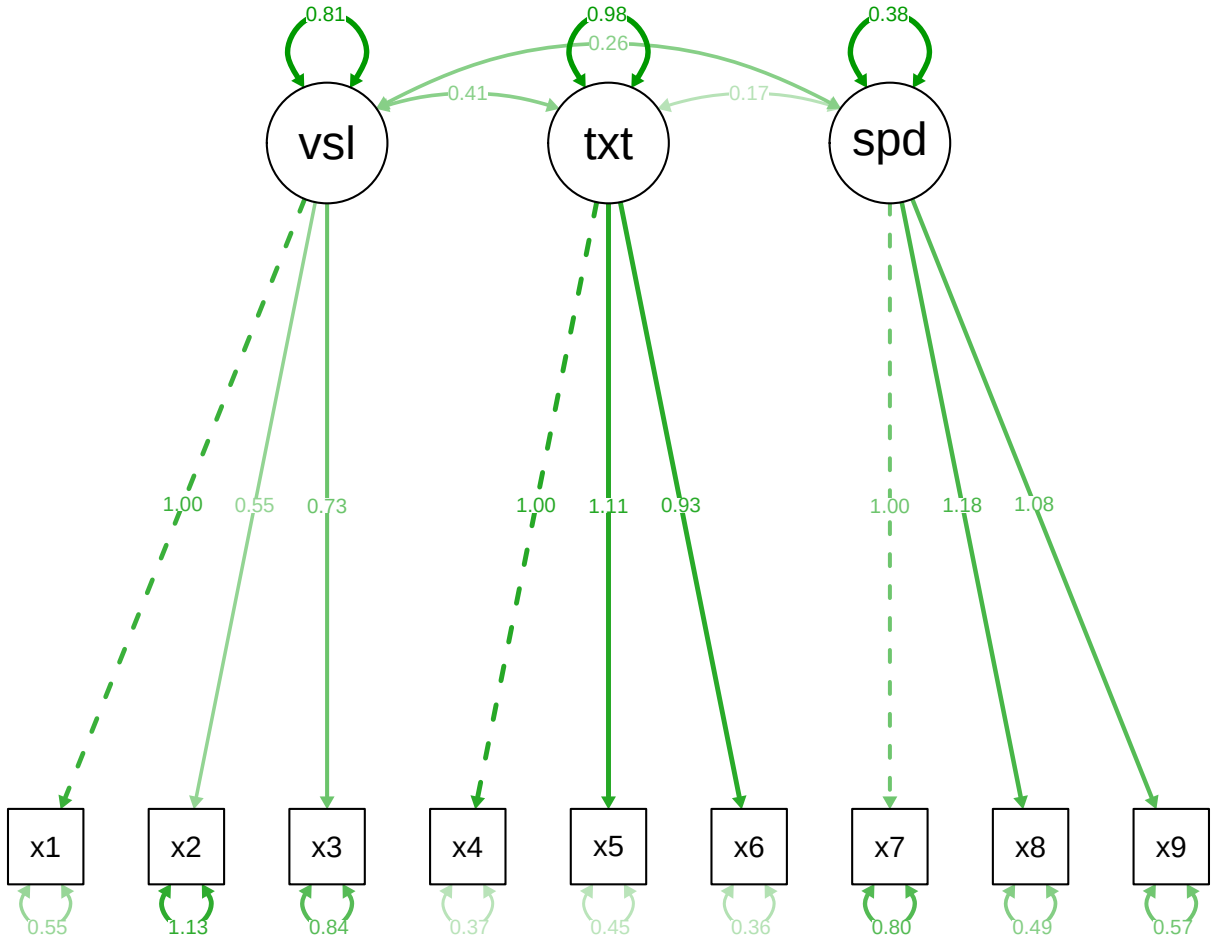
```
semPaths(fit)
```



```
semPaths(fit, "Standardized", "Estimates")
```



```
semPaths(fit, "std", "est")
```



semPaths

`semPaths` has quite a lot of arguments:

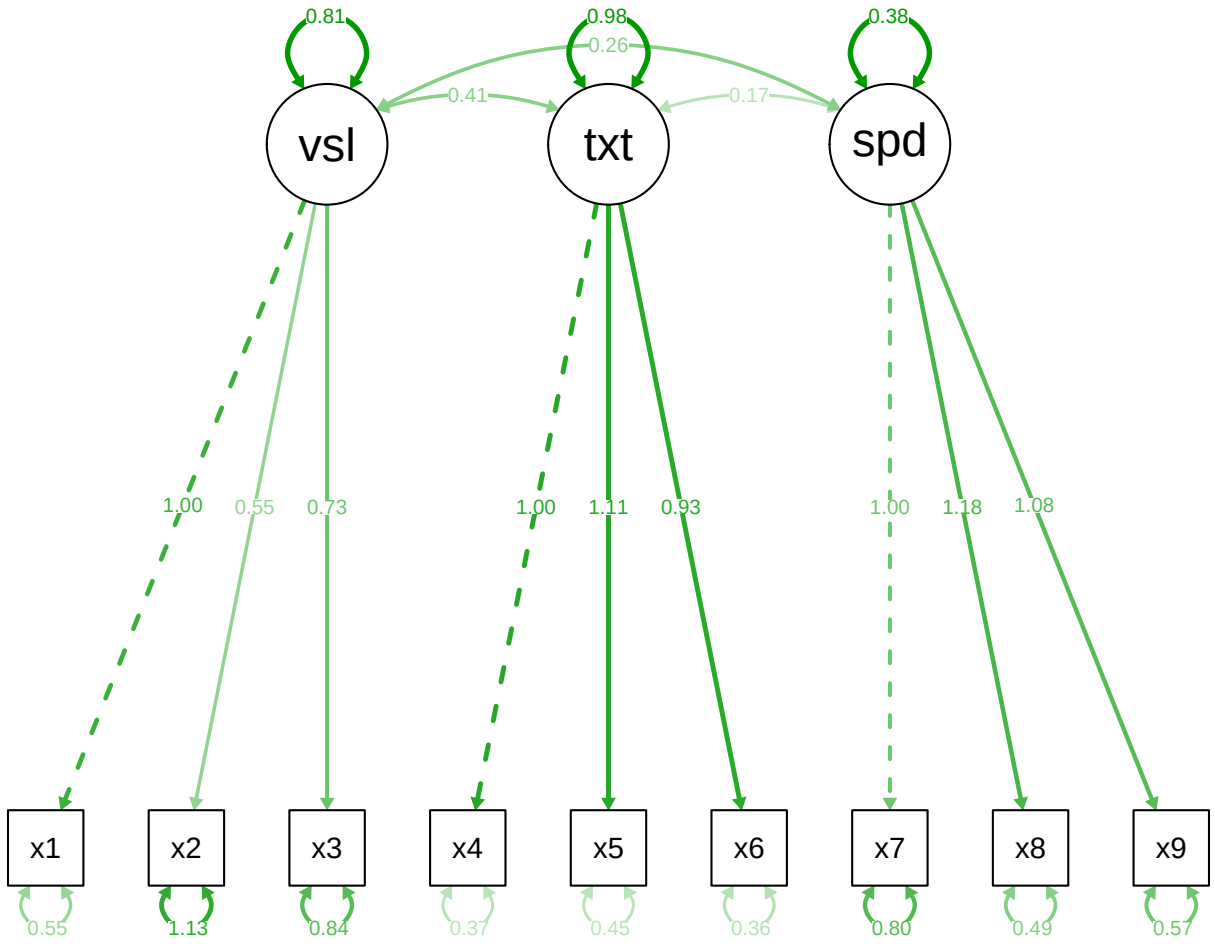
style, layout, intercepts, residuals, thresholds, rotation, curve, curvature, nCharNodes, nCharEdges, sizeMan, sizeLat, sizeInt, sizeMan2, sizeLat2, sizeInt2, shapeMan, shapeLat, shapeInt, ask, mar, title, title.color, title.adj, title.line, title.cex, include, combineGroups, manifests, latents, groups, color, residScale, gui, allVars, edge.color, reorder, structural, ThreshAtSide, thresholdColor, thresholdSize, fixedStyle, freeStyle, as.expression, optimizeLatRes, inheritColor, levels, nodeLabels, edgeLabels, pastel, rainbowStart, intAtSide, springLevels, nDigits, exoVar, exoCov, centerLevels, panelGroups, layoutSplit, measurementLayout, subScale, subScale2, subRes, subLinks, modelOpts, curveAdjacent, edge.label.cex, cardinal, equalizeManifests, covAtResiduals, bifactor, optimPoints

semPaths

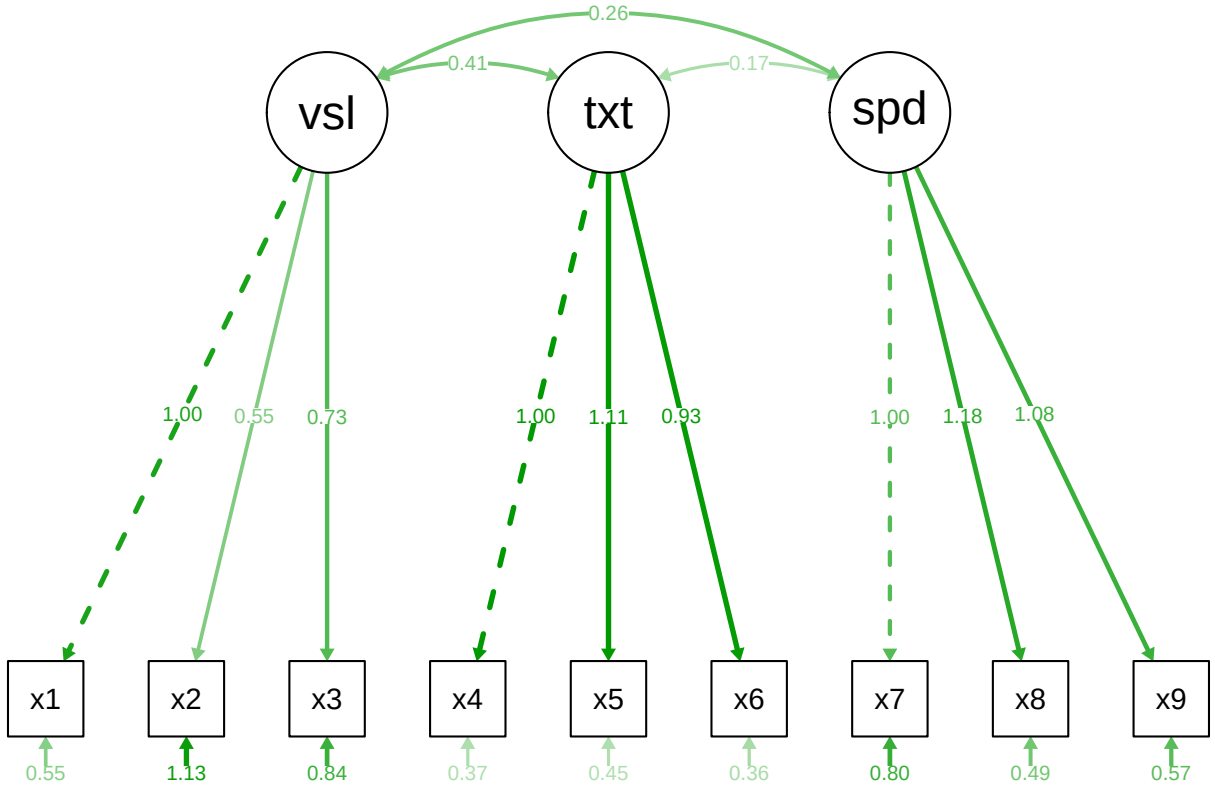
And even more via the `qgraph` backend:

edge.width, node.width, node.height, esize, asize, minimum, maximum, cut, details, mar, filetype, filename, width, height, normalize, DoNotPlot, plot, rescale, label.cex, label.color, borders, border.color, border.width, polygonList, vTrans, label.prop, label.norm, label.scale, label.font, posCol, negCol, unCol, colFactor, trans, fade, loop, curvePivot,curvePivotShape, edge.label.bg, edge.label.position, edge.label.font, layout.par, bg, bgcontrol, bgres, pty, font, arrows, arrowAngle, asize, open, weighted, XKCD, . . .

```
semPaths(fit, "std", "est", style = "mx")
```



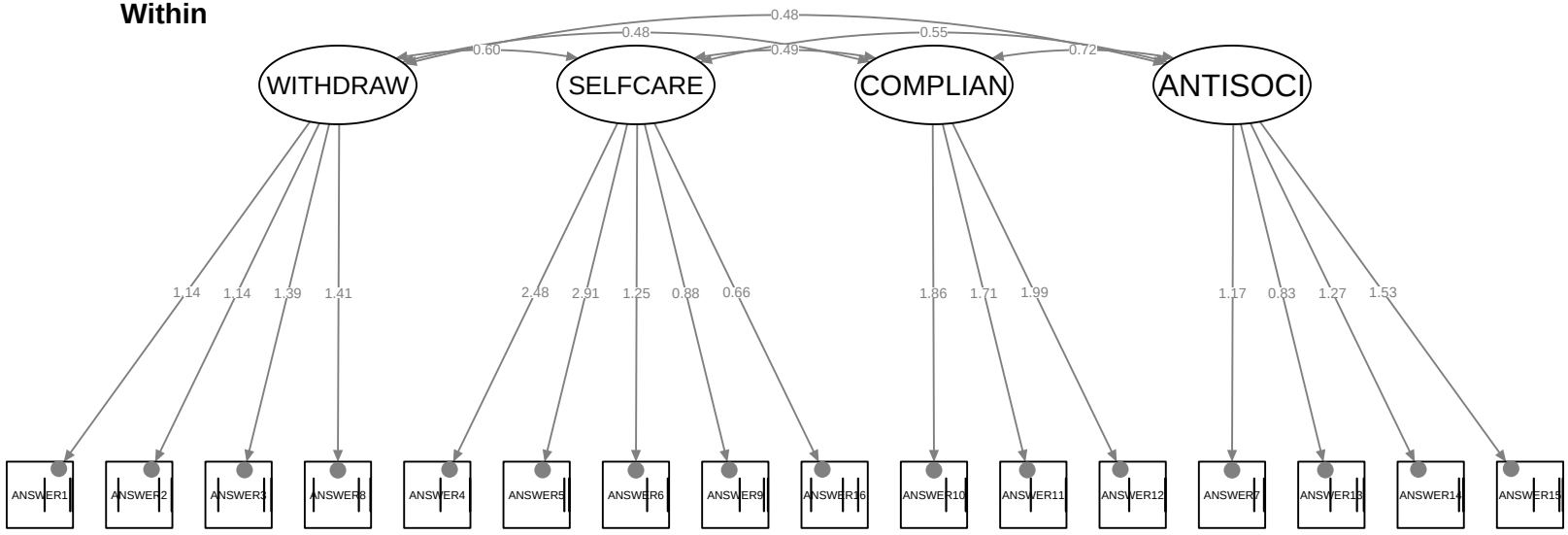
```
semPaths(fit, "std", "est", style = "lisrel")
```



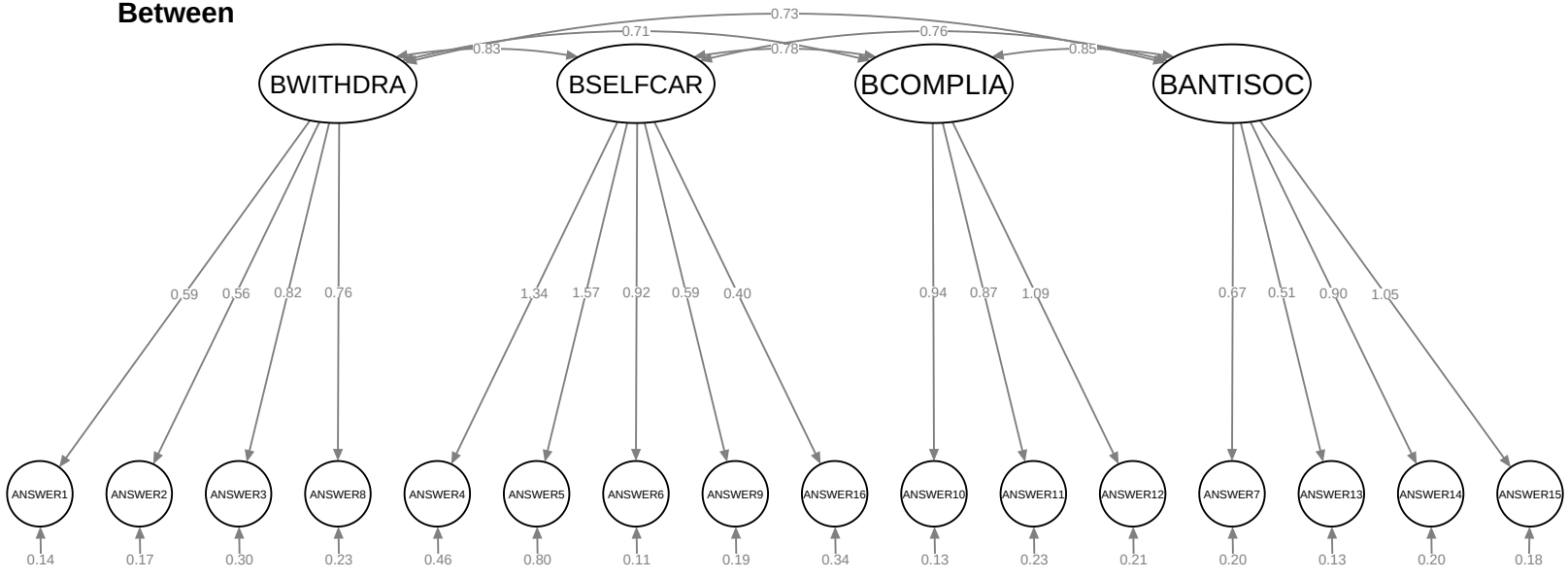
Multi-level

```
semPaths(file.choose(), "model", "estimates",
    style = "lisrel", curve = 0.8, nCharNodes = 0,
    sizeLat = 12, sizeLat2 = 6, title = TRUE,
    mar = c(5, 1, 5, 1), curvePivot = FALSE,
    edge.label.cex = 0.5)
```

Within



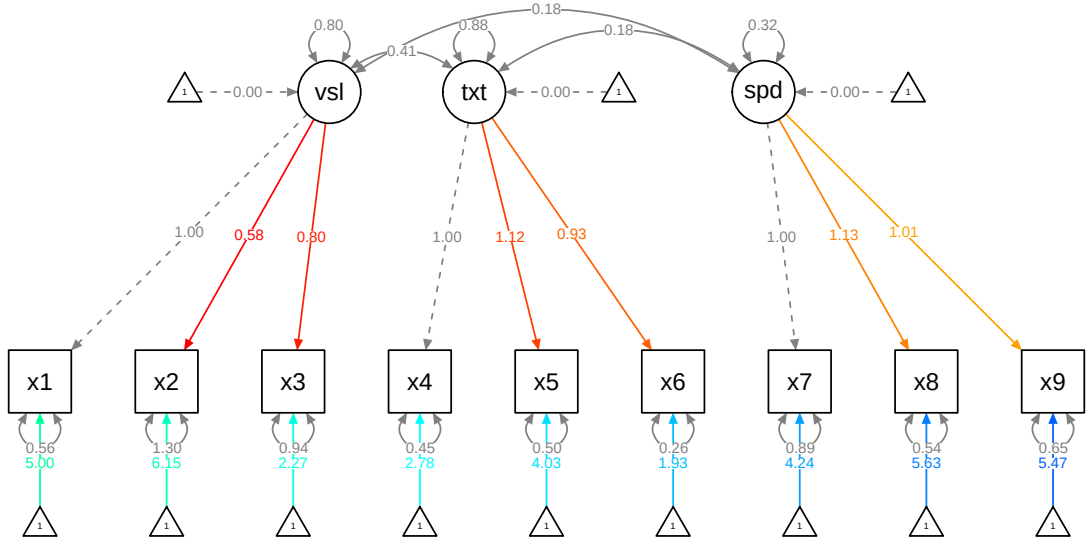
Between

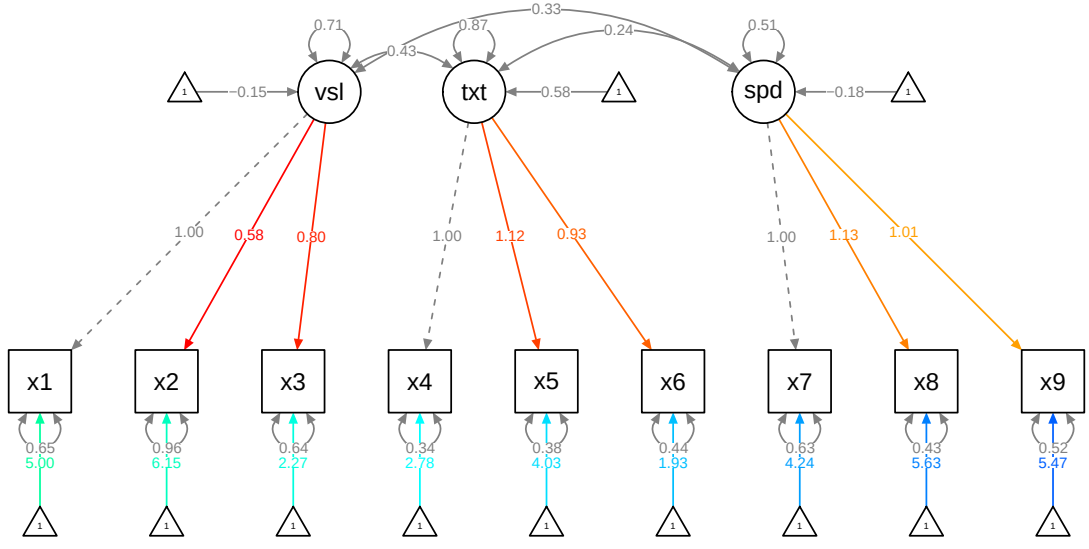


Constraints

```
library("semTools")
fits <- example(measurementInvariance)
semPaths(fits$value$fit.intercepts, "equality",
    "estimates", sizeLat = 5, title = FALSE,
    ask = FALSE, levels = c(1, 2, 4), edge.label.cex = 0.5,
    mar = c(0.1, 0.1, 0.1, 0.1))
```

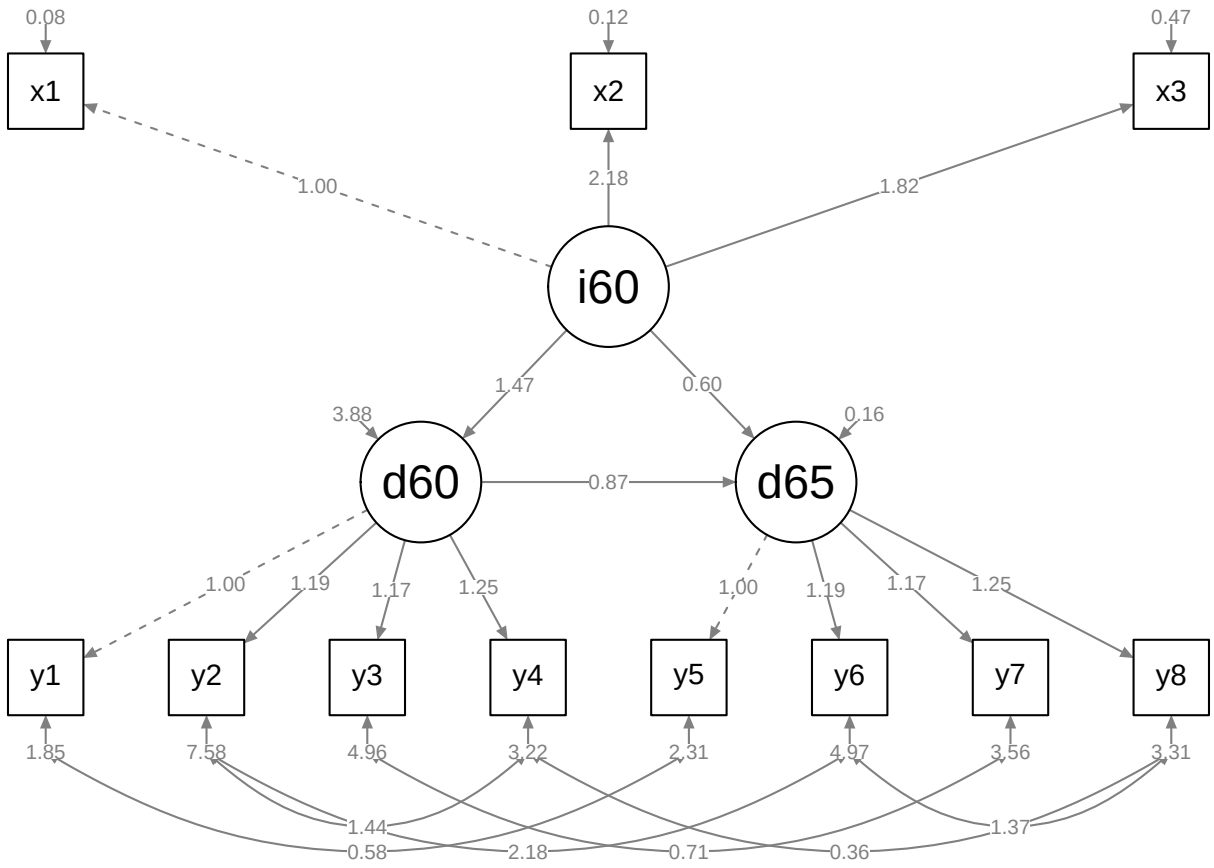
Constraints





Structural Models

```
# lavaan sem example:
example(sem)
semPaths(fit, "model", "est", style = "lisrel")
```



Layout Algorithms

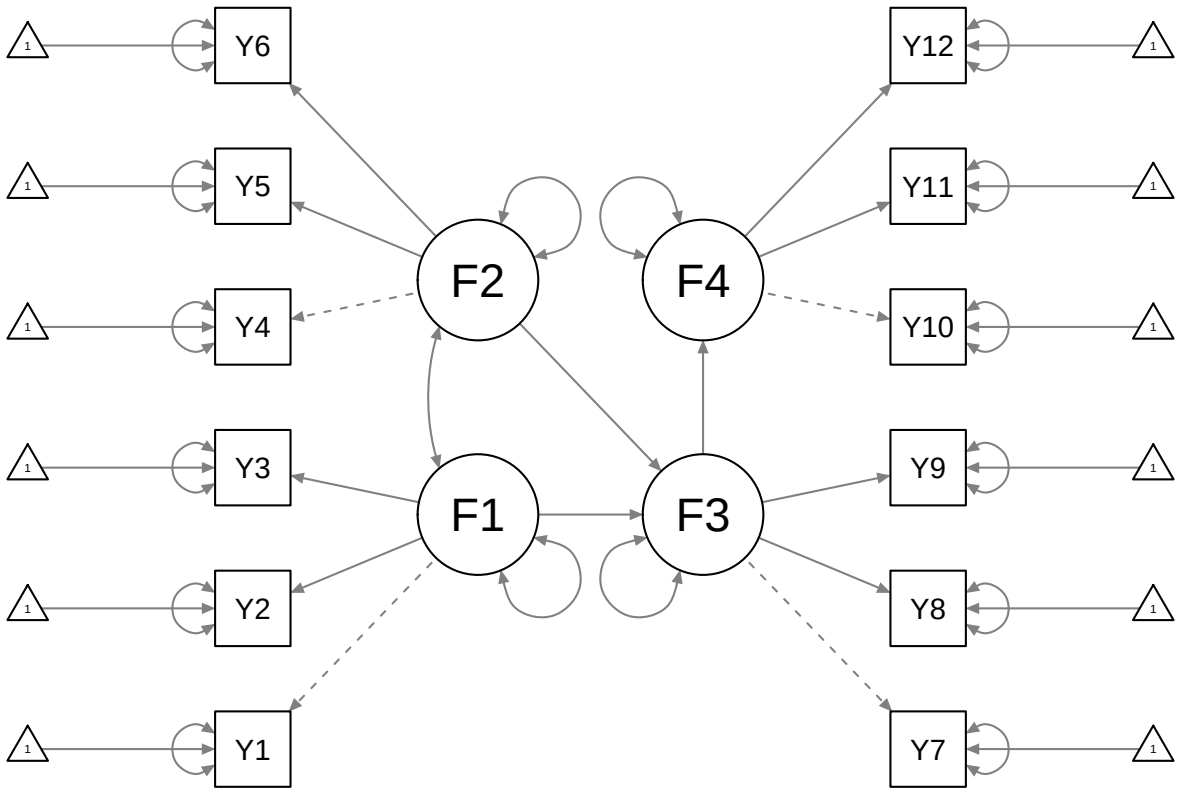
- `semPaths` can use several tree-like layout algorithms

tree	Based on LISREL (Jöreskog & Sörbom, 1996)
tree2	Variation of the Reingold-Tilford algorithm (Reingold & Tilford, 1981)
tree3	Variation of Boker, McArdle, and Neale (2002)

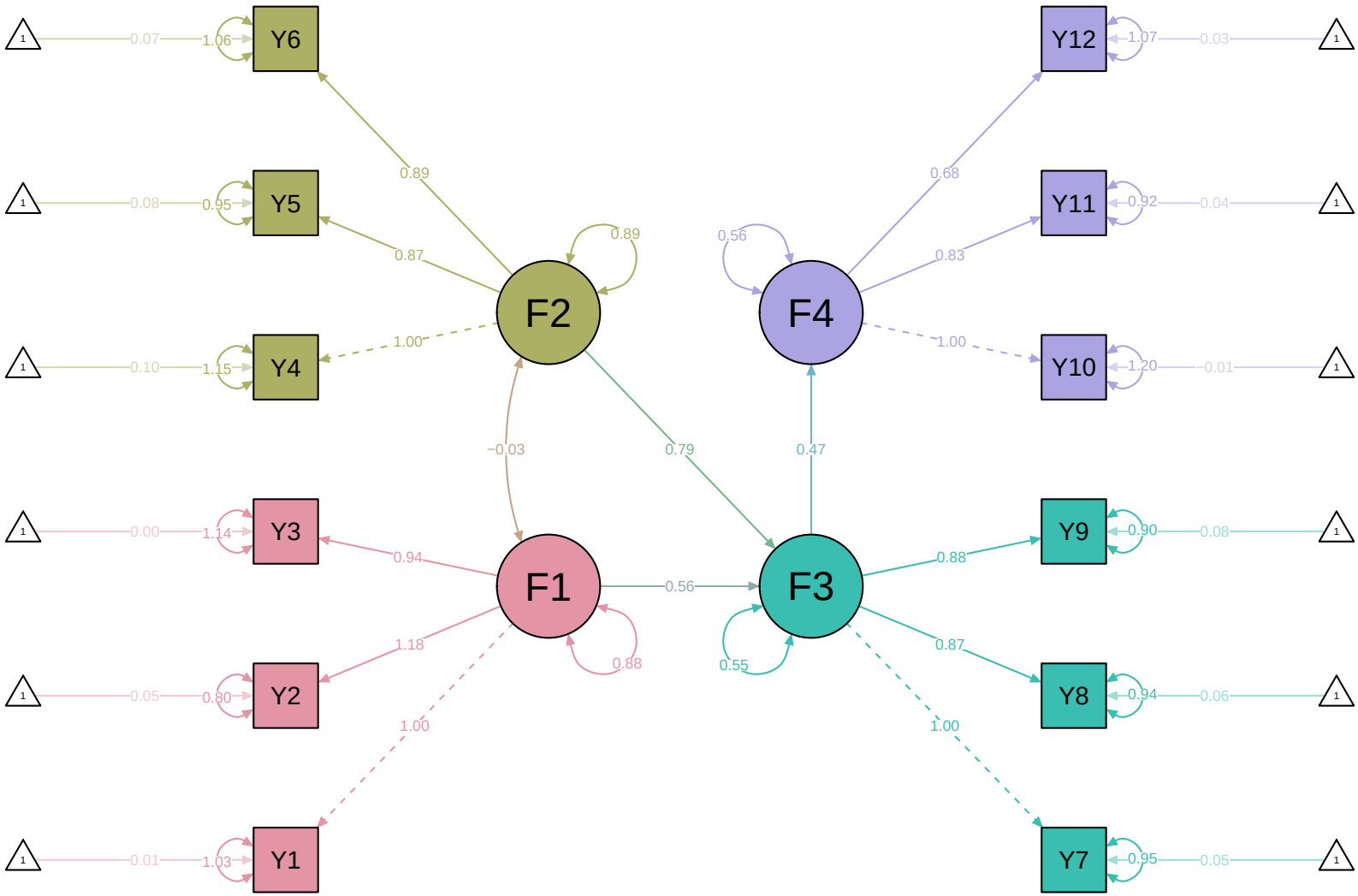
- Exogenous variables on top, endogenous variables at the bottom
  - Can be rotated
- These layouts can be circularized (`circle`, `circle2` and `circle3`)
- Alternatively any igraph algorithm can be used
- `layoutSplit` can be used to split layout of measurement models and structural model

```r
# Example 5.25 from mplus user guide:
l <- "http://www.statmodel.com/usersguide/chap5/ex5.11.out"
download.file(l, modfile <- tempfile(fileext = ".out"))
Model <- semPlotModel(modfile)
```

```
semPaths(Model, rotation = 2)
```

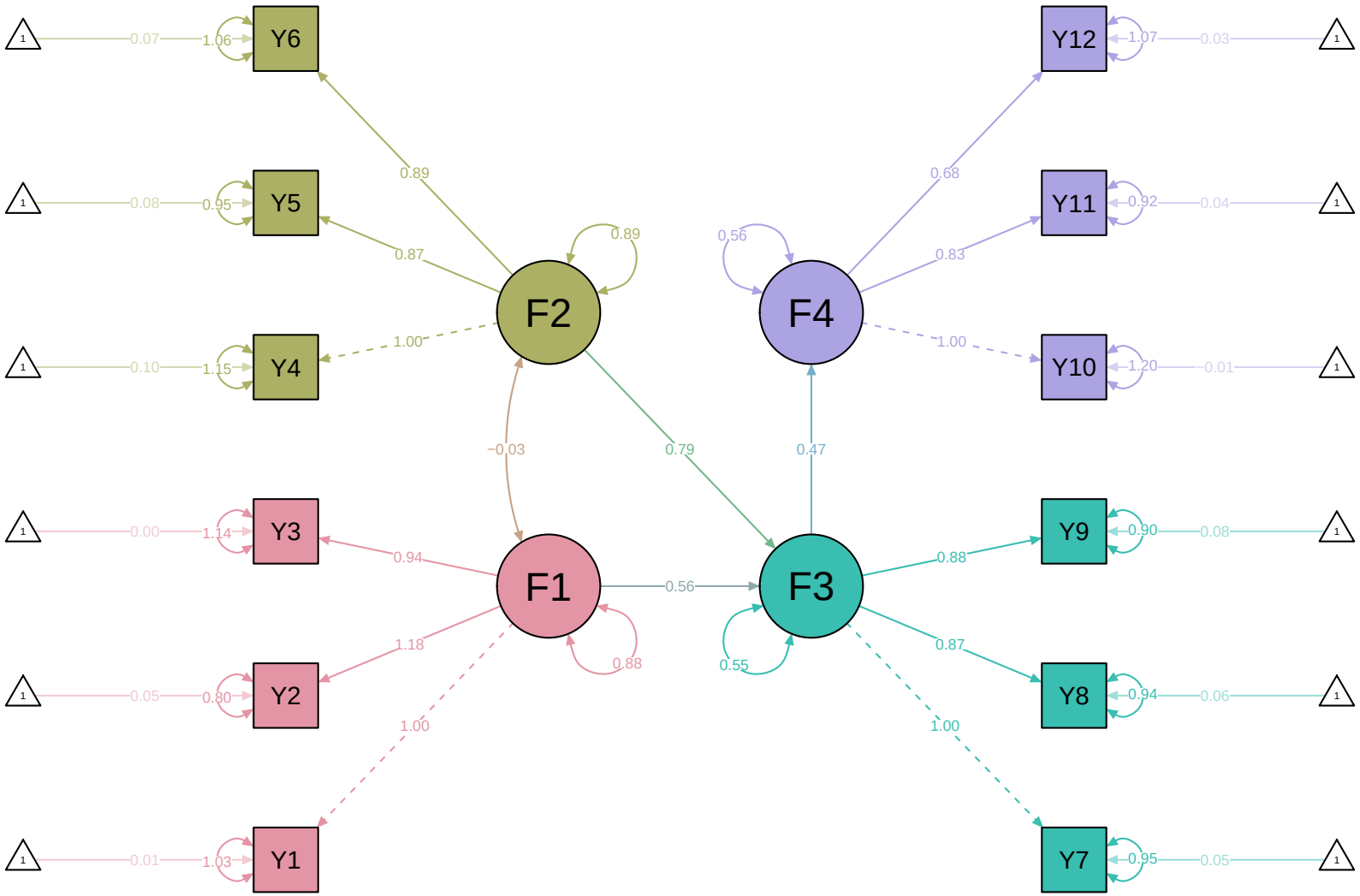


```
semPaths(Model, "col", "est", rotation = 2,
    groups = "latents", pastel = TRUE,
    edge.label.cex = 0.5, intercepts = TRUE,
    mar = c(1, 1, 1, 1))
```



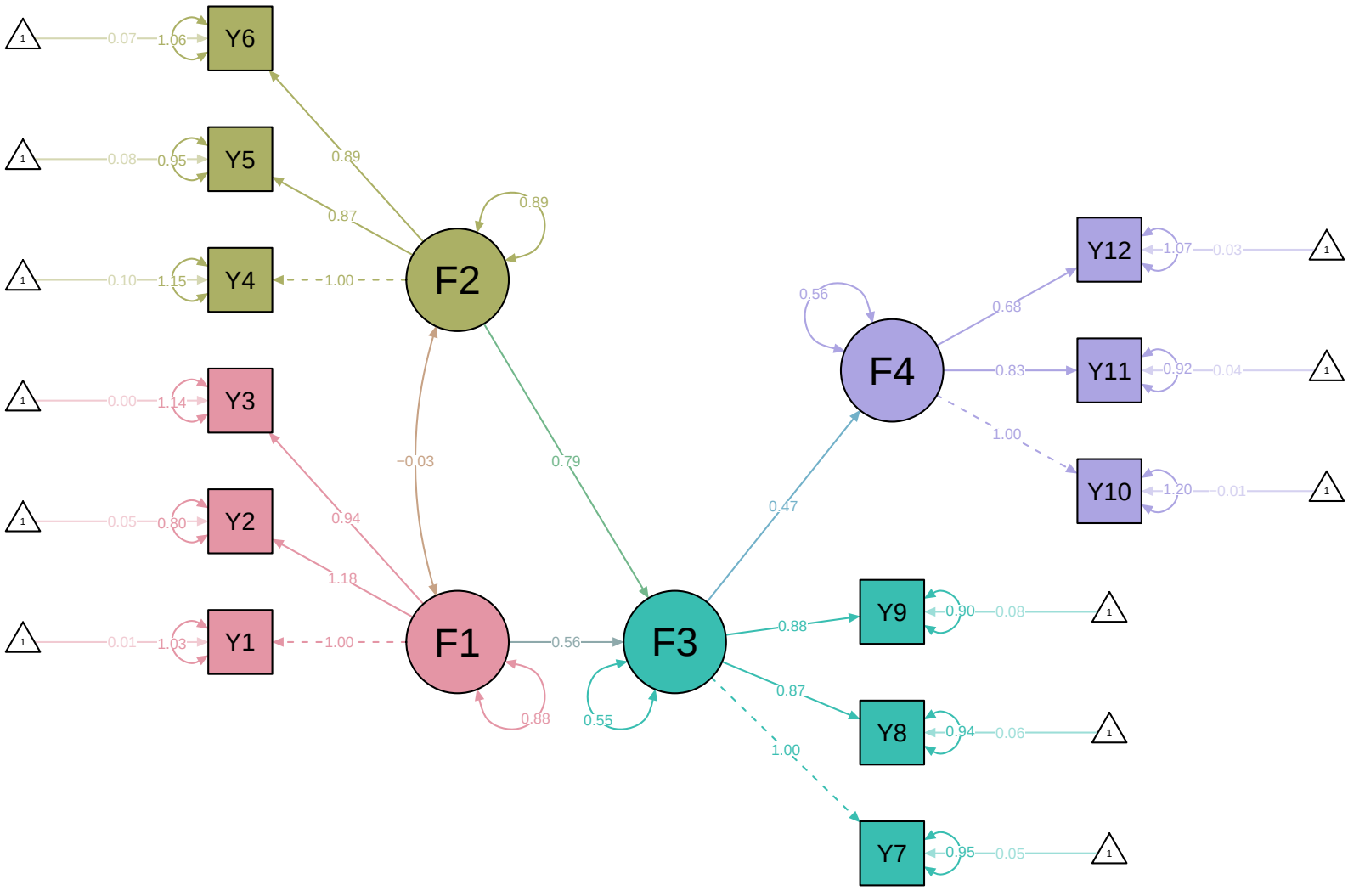
LISREL style layout

```
semPaths(<...>, layout = "tree")
```



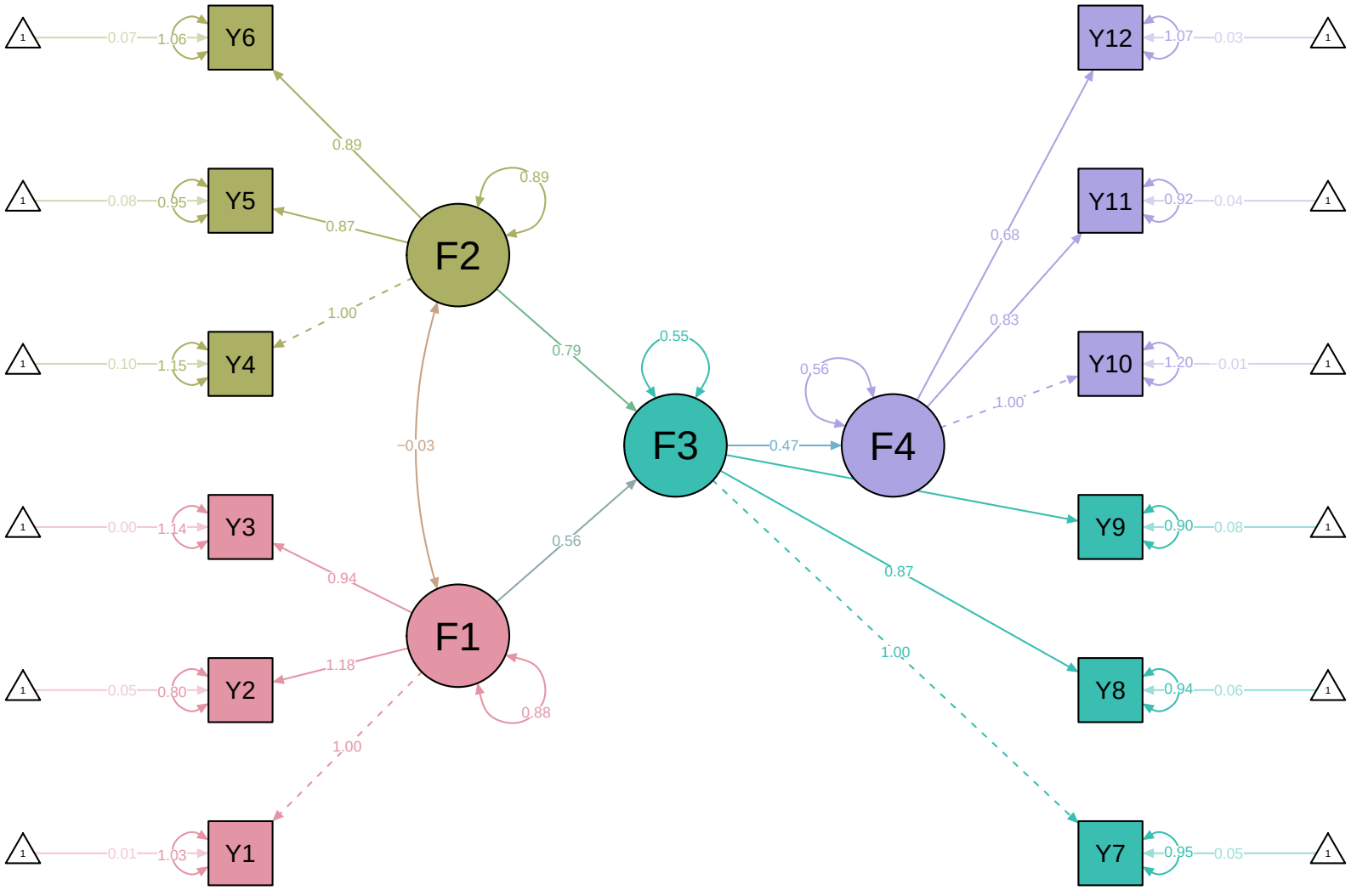
Reingold-Tilford based layout

```
semPaths(<...>, layout = "tree2", centerLevels = FALSE)
```



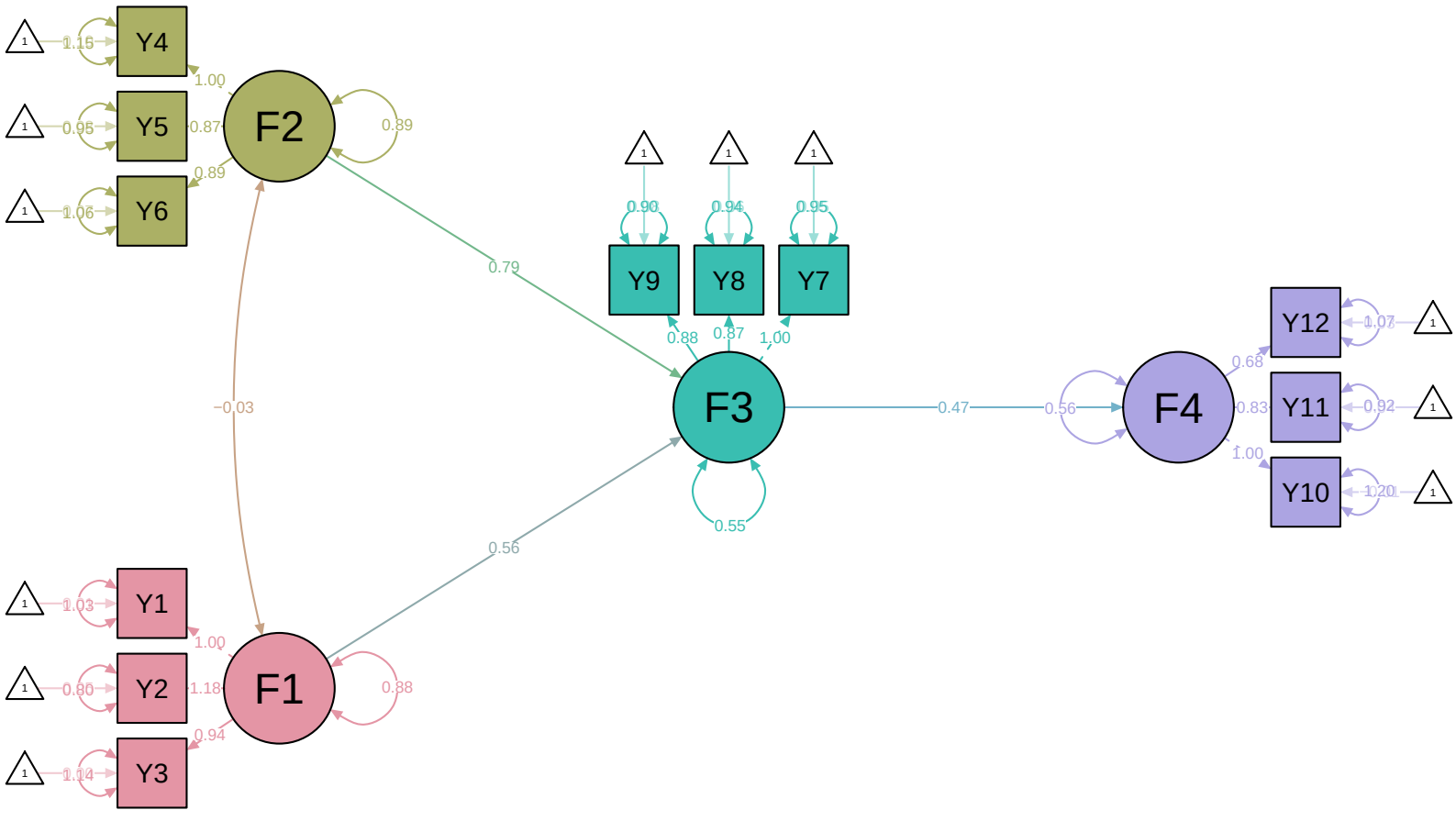
Boker-McArdle-Neale based layout

```
semPaths(<...>, layout = "tree3", optimizeLatRes = TRUE)
```

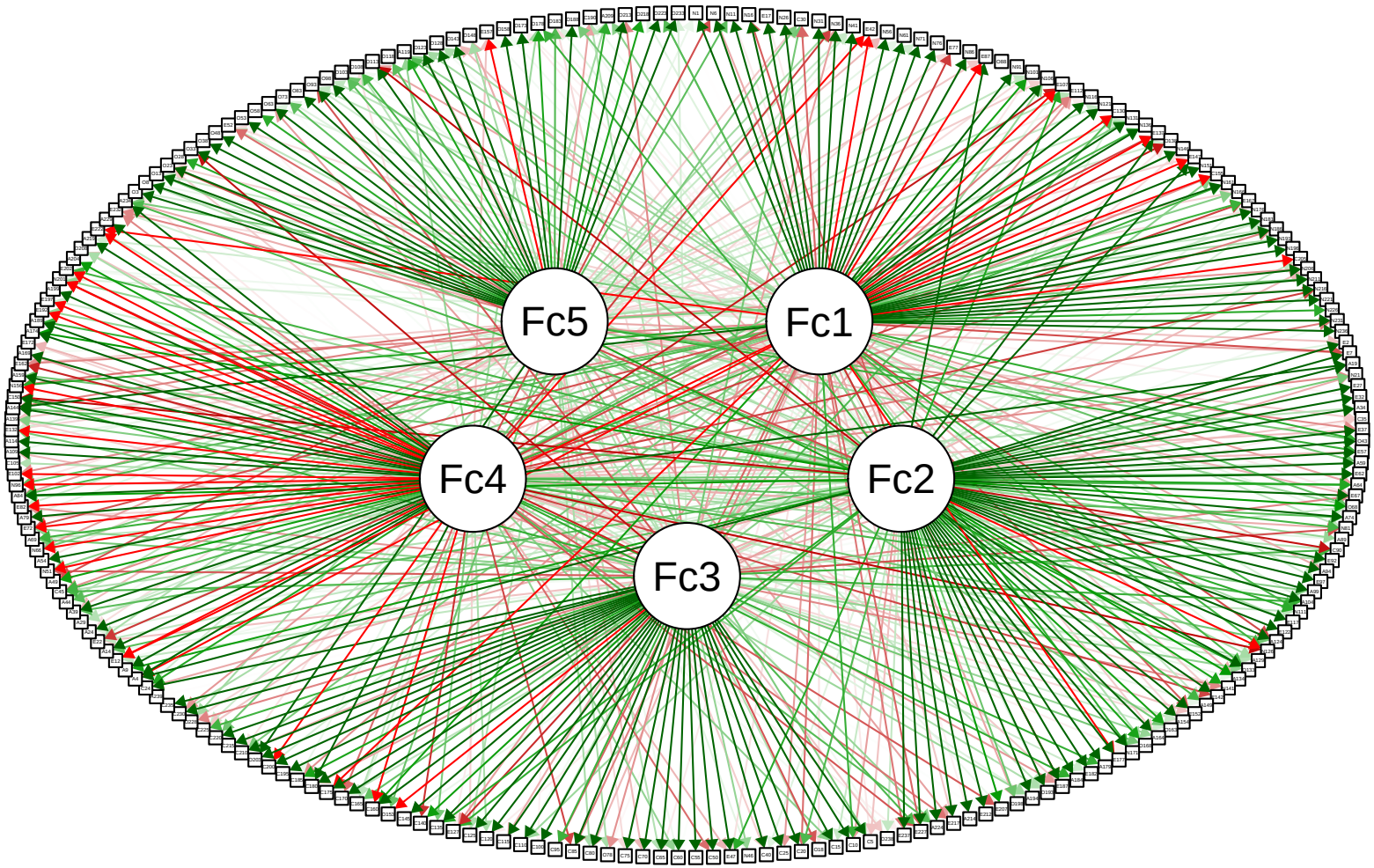


Split measurement and structural models

```
semPaths(<...>, layout = "tree3", layoutSplit = TRUE)
```



```r
library("qgraph")
data(big5)
res <- factanal(big5, 5, rotation = "promax")
semPaths(res, "standardized", "hide", residuals = FALSE,
    sizeMan = 1, mar = c(1, 1, 1, 1), NCharNodes = 0,
    layout = "circle")
```

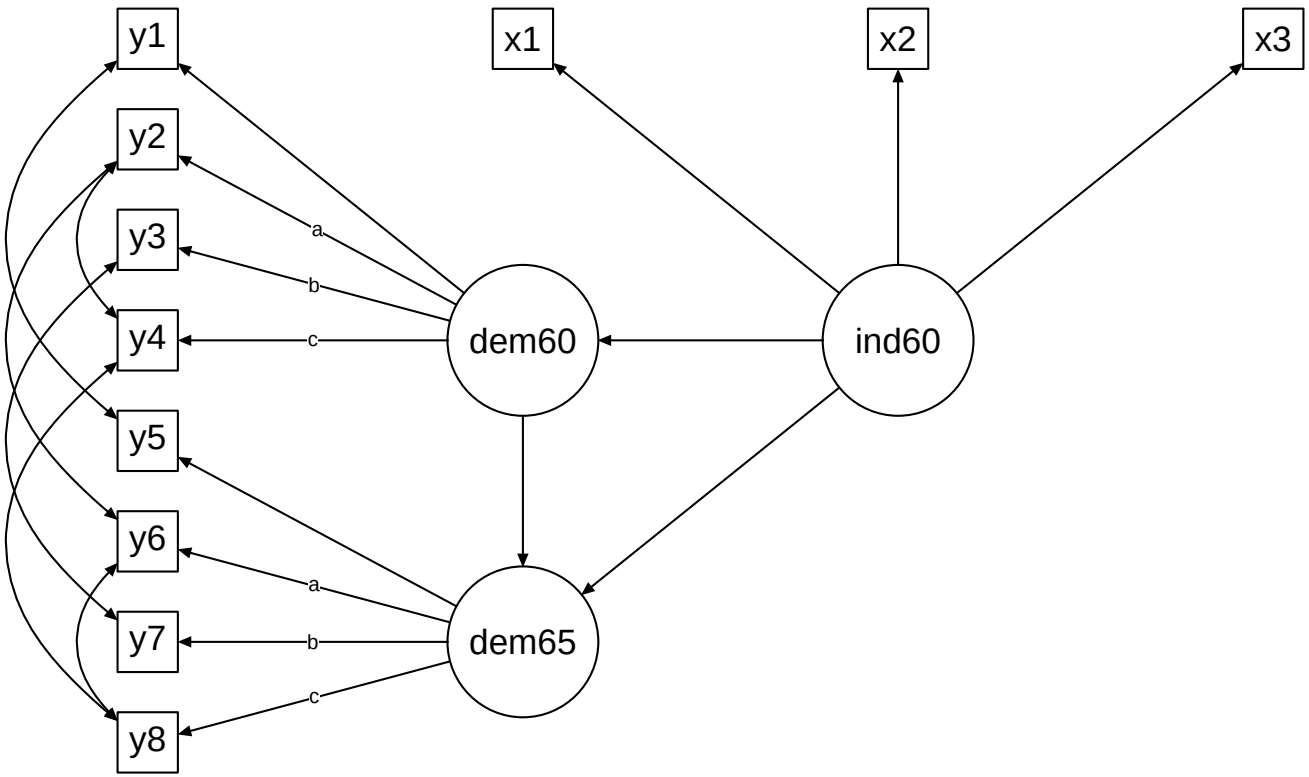


Manual specification

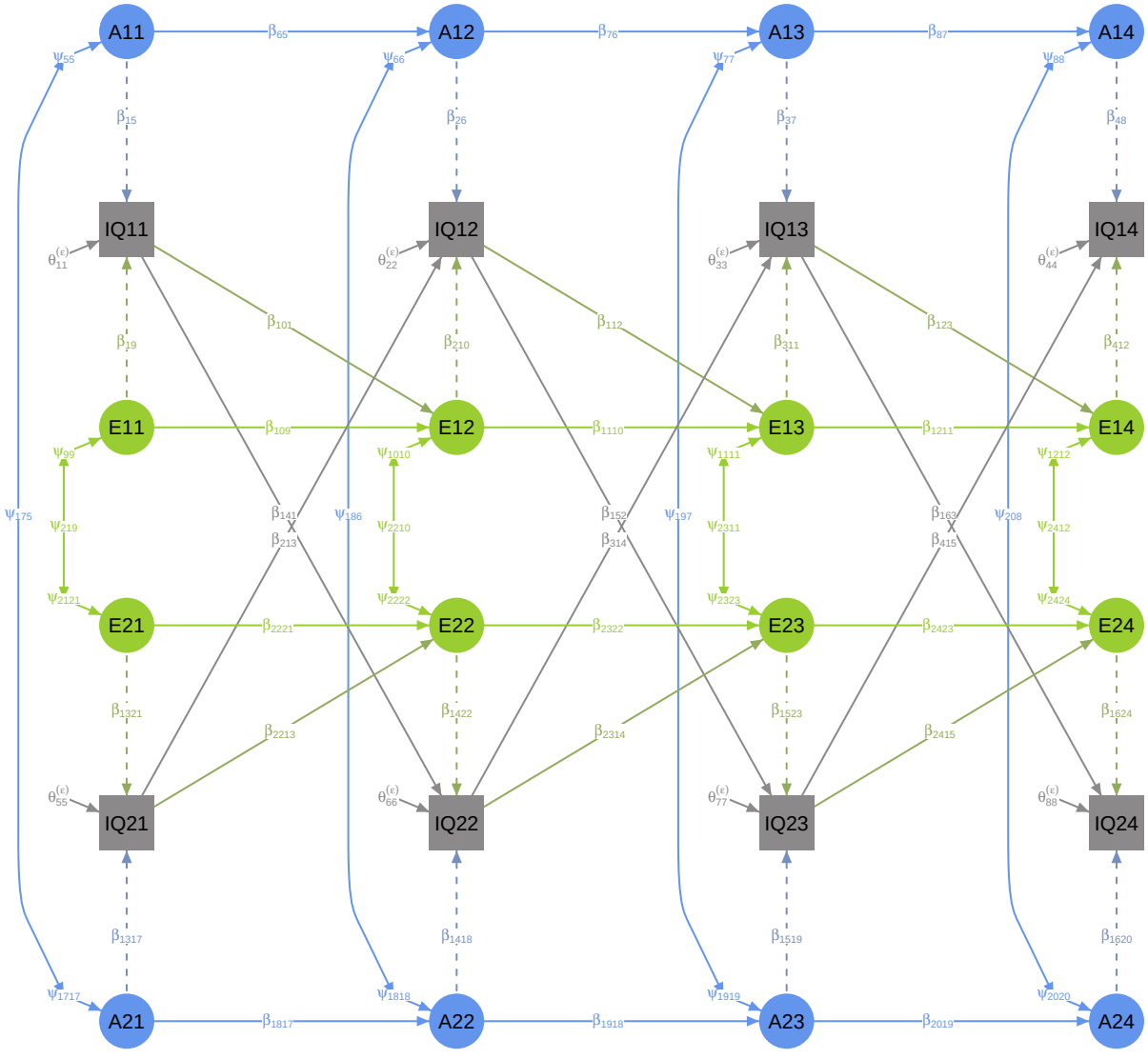
```
library('lavaan')

example(sem)

L <- matrix(
  c(
    "y1", "y2", "y3", "y4",     "y5", "y6", "y7",    "y8",
    "x1", NA,    NA,    "dem60", NA,     NA,   "dem65", NA,
    "x2", NA,    NA,    "ind60", NA,     NA,    NA,      NA,
    "x3", NA,    NA,     NA,      NA ,    NA,    NA,      NA),
  ,4 )
```

```r
Graph <- semPaths(fit,
                  layout=L,
                  nCharNodes=0,
                  edge.color="black",
                  label.scale=FALSE,
                  label.cex=1.0,
                  residuals=FALSE,
                  fixedStyle=1,
                  freeStyle=1,
                  exoVar=FALSE,
                  sizeMan=4,
                  sizeLat=10,
                  DoNotPlot = TRUE
)

Graph$graphAttributes$Edges$curve <-
    ifelse(Graph$Edgelist$bidir, -2, 0)
```

```
plot(Graph)
```



Model by Janneke de Kort



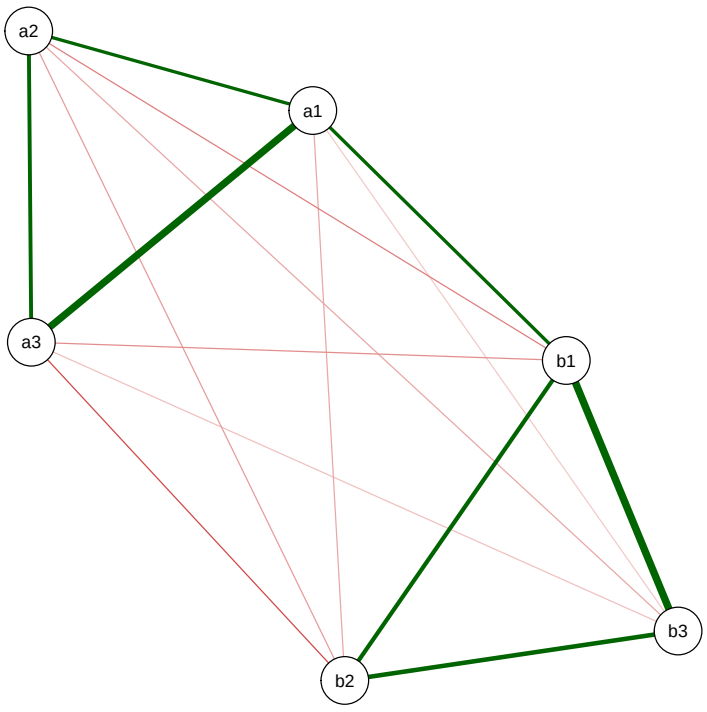
`semCors()` can be used to plot implied and observed covariances using the **qgraph** framework (Epskamp et al., 2012). For example:

```
library("lavaan")

# Simulate 2 factor model with correlated residual:
Mod <- '
A =~ 1*a1 + 0.6*a2 + 0.8*a3
B =~ 1*b1 + 0.7*b2 + 0.9*b3
a1 ~~ 1*b1
A ~~ -0.3* B
'

set.seed(5)
Data <- simulateData(Mod)

# Fit regular 2 factor model:
Mod <- '
A =~ a1 + a2 + a3
B =~ b1 + b2 + b3
'
fit <- cfa(Mod, data=Data)
```

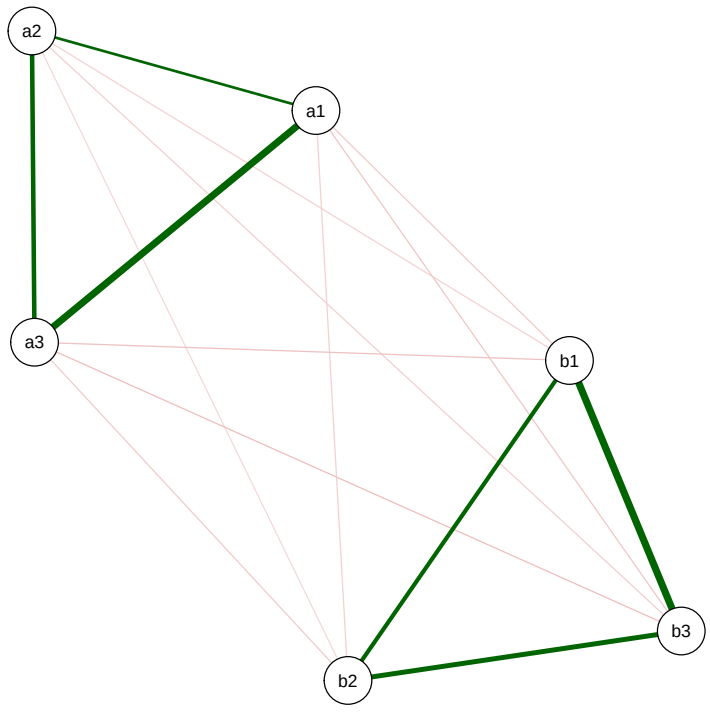
Fit it in lavaan and look at the covariance matrices:

```
semCors(fit, layout = "spring", cut = 0.3,
    esize = 20, titles = TRUE)
```

Observed



Implied



The `modelMatrices()` function can be used to obtain a list of all matrices in one of three modeling frameworks:

- RAM
- LISREL
- Mplus

```r
names(modelMatrices(fit, "ram"))
```

```
## [1] "A" "S" "F"
```

```r
names(modelMatrices(fit, "lisrel"))
```

```
##  [1] "LY" "TE" "PS" "BE" "LX" "TD"
##  [7] "PH" "GA" "TY" "TX" "AL" "KA"
```

```r
names(modelMatrices(fit, "mplus"))
```

```
## [1] "Nu"     "Lambda" "Theta"
## [4] "Kappa"  "Alpha"  "Beta"
## [7] "Gamma"  "Psi"
```

modelMatrices

```
str(modelMatrices(fit, "ram")$A)

## List of 1
##  $ :List of 4
##   ..$ est  : num [1:8, 1:8] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   ..$ std  : num [1:8, 1:8] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   ..$ par  : num [1:8, 1:8] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   ..$ fixed: logi [1:8, 1:8] FALSE FALSE FALSE FALSE FALSE
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
##   .. .. ..$ : chr [1:8] "a1" "a2" "a3" "b1" ...
```

```r
modelMatrices(fit, "ram")$A[[1]]$est
```

```
##    a1 a2 a3 b1 b2 b3      A     B
## a1  0  0  0  0  0  0 1.0000 0.000
## a2  0  0  0  0  0  0 0.7335 0.000
## a3  0  0  0  0  0  0 1.0390 0.000
## b1  0  0  0  0  0  0 0.0000 1.000
## b2  0  0  0  0  0  0 0.0000 0.765
## b3  0  0  0  0  0  0 0.0000 1.012
## A   0  0  0  0  0  0 0.0000 0.000
## B   0  0  0  0  0  0 0.0000 0.000
```

The `semMatrixAlgebra()` function makes extracting matrices easier:

```
semMatrixAlgebra(fit, A)

## model set to 'ram'

##    a1 a2 a3 b1 b2 b3      A     B
## a1  0  0  0  0  0  0 1.0000 0.000
## a2  0  0  0  0  0  0 0.7335 0.000
## a3  0  0  0  0  0  0 1.0390 0.000
## b1  0  0  0  0  0  0 0.0000 1.000
## b2  0  0  0  0  0  0 0.0000 0.765
## b3  0  0  0  0  0  0 0.0000 1.012
## A   0  0  0  0  0  0 0.0000 0.000
## B   0  0  0  0  0  0 0.0000 0.000
```

Note how using the term `A` caused the function to automatically identify we were interested in the RAM model.

`semMatriAlgebra()` can also be used to easily perform algebraic computations:

```
semMatrixAlgebra(fit, Lambda %*% Psi %*% t(Lambda) + Theta)

## model set to 'mplus'

##           a1       a2       a3       b1       b2       b3
## a1  2.02879  0.60113  0.85151 -0.12520 -0.09578 -0.12674
## a2  0.60113  1.52291  0.62456 -0.09183 -0.07025 -0.09296
## a3  0.85151  0.62456  1.63260 -0.13008 -0.09951 -0.13168
## b1 -0.12520 -0.09183 -0.13008  1.95964  0.66839  0.88447
## b2 -0.09578 -0.07025 -0.09951  0.66839  1.53194  0.67661
## b3 -0.12674 -0.09296 -0.13168  0.88447  0.67661  1.78813
```

Also works for multi-group analyses:

```
l <- "http://www.statmodel.com/examples/continuous/cont12.html
download.file(l, modfile <- tempfile(fileext = ".out"))

semMatrixAlgebra(modfile, Theta)

## model set to 'mplus'

## Reading model:  C:\Users\sacha\AppData\Local\Temp\RtmpeqxVk
## [[1]]
##       Y6    Y7    Y8    Y9
## Y6 0.354 0.000 0.000 0.000
## Y7 0.000 0.268 0.000 0.000
## Y8 0.000 0.000 1.374 0.000
## Y9 0.000 0.000 0.000 2.528
##
## [[2]]
##       Y6    Y7    Y8    Y9
## Y6 0.354 0.000 0.000 0.000
## Y7 0.000 0.268 0.000 0.000
## Y8 0.000 0.000 1.374 0.000
## Y9 0.000 0.000 0.000 2.528
```

`semSyntax` can be used to translate any input to `semPlot` into **lavaan** codes. This has two advantages:

- Easily fit a model based on an output file in **lavaan**
- Simulate data based on an estimated model using **lavaan**'s `simulateData`

Translating **lavaan** syntax to **MPlus** syntax can be attempted using `lavaan:::lav2mplus`. **sem** is also supported but a bit bugged at the moment. Mail me for a **lavaan** to **OpenMx** translator.

Translate **MPlus** to **lavaan**:

```
l <- "http://www.statmodel.com/usersguide/chap5/ex5.1.out"
download.file(l, modfile <- tempfile(fileext = ".out"))
Model <- semPlotModel(modfile)

## Reading model:  C:\Users\sacha\AppData\Local\Temp\RtmpeqxVk

lavMod <- semSyntax(Model)

##
## Model <- '
## F1 =~ 1*Y1
## F1 =~ Y2
## F1 =~ Y3
## F2 =~ 1*Y4
## F2 =~ Y5
## F2 =~ Y6
## F2 ~~ F1
## Y1 ~ 1
## Y2 ~ 1
## Y3 ~ 1
## Y4 ~ 1
## Y5 ~ 1
## Y6 ~ 1
```

Simulate data:

```
l <- "http://www.statmodel.com/usersguide/chap5/ex5.1.out"
download.file(l, modfile <- tempfile(fileext = ".out"))
Model <- semPlotModel(modfile)
lavMod <- semSyntax(Model, allFixed = TRUE)
```

Simulate data:

```
library("lavaan")
head(simulateData(lavMod))
```

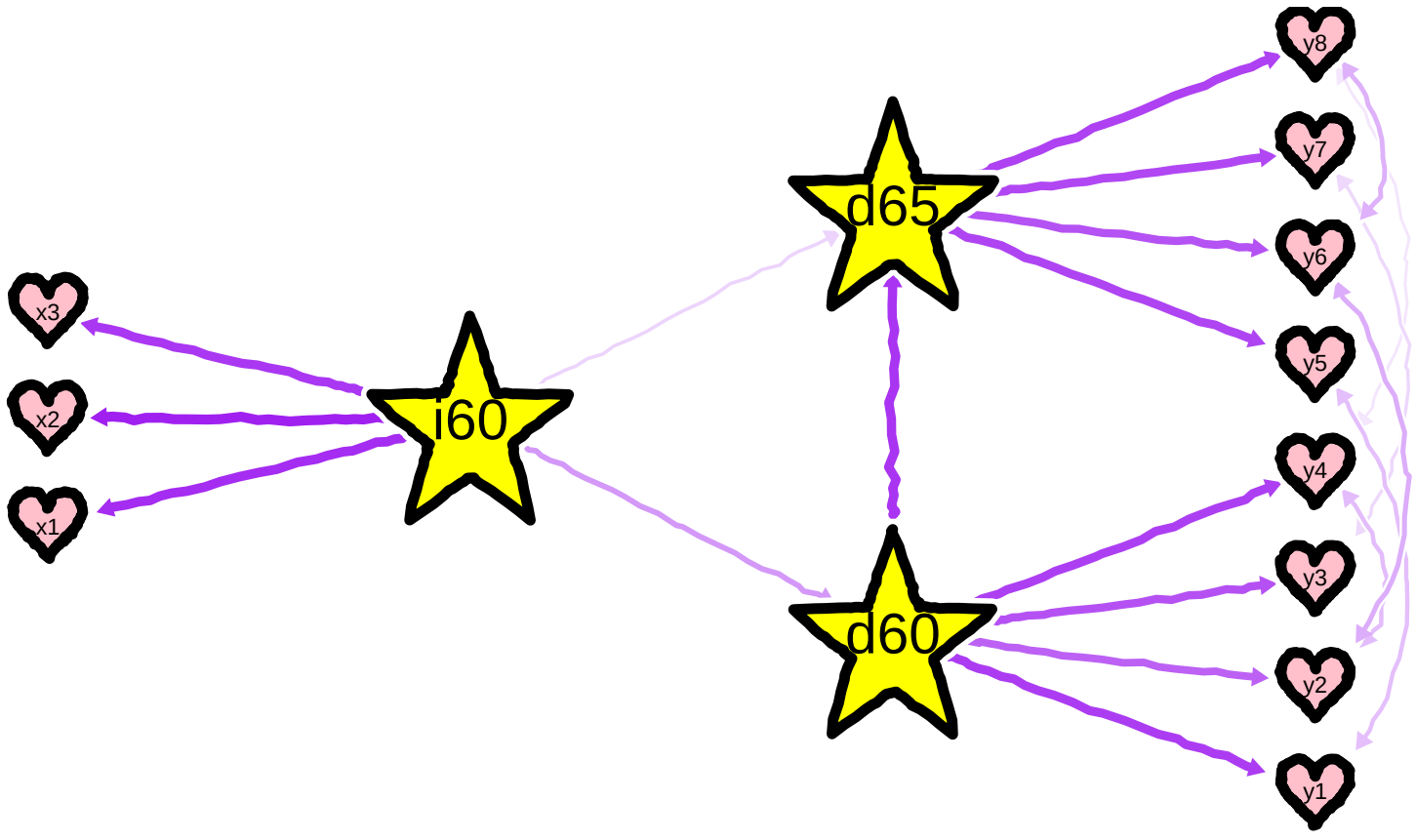
```
##         Y1       Y2       Y3      Y4      Y5      Y6
## 1 -0.1812 -0.86023 -0.26249  0.8436  1.3738 -0.2065
## 2  0.4026 -1.42322 -0.03974  0.6176  0.5889  0.6993
## 3  1.2055  0.37841  1.44397  0.7376  0.9466 -0.8903
## 4  2.1490 -0.67511  0.07165  0.1718 -0.4993 -2.1682
## 5  0.3397 -0.09025 -0.06618 -1.2264  0.0610 -1.2726
## 6 -1.5069 -0.81482 -1.58714  1.1065 -0.4947  0.2997
```

Future directions

- (Better) support for:
  - **Onyx**
  - **Amos**
  - **EQS**
  - **lava**
- Extension to different models:
  - LKA
  - IRT
  - Bayesian models

In the spirit of Valentine

```
library("lavaan")
example(sem)
semPaths(fit, "std", "hide", sizeLat = 15, shapeLat = "star", shapeMan = "heart",
    col = list(man = "pink", lat = "yellow"), residuals = FALSE, borders = FALSE,
    edge.color = "purple", XKCD = TRUE, edge.width = 2, rotation = 2, layout = "tree2",
    fixedStyle = 1, mar = c(1, 3, 1, 3))
```



Thank you for your attention!

References I

- Boker, S. M., McArdle, J., & Neale, M. (2002). An algorithm for the hierarchical organization of path diagrams and calculation of components of expected covariance. *Structural Equation Modeling*, *9*(2), 174–194.
- Boker, S. M., Neale, M., Maes, H., Wilde, M., Spiegel, M., Brick, T., . . . Fox, J. (2011). OpenMx: an open source extended structural equation modeling framework. *Psychometrika*, *76*(2), 306–317.
- Epskamp, S. (2013). *lisrelToR: import output from LISREL into R*. R package version 0.1.4. Retrieved from http://CRAN.R-project.org/package=lisrelToR
- Epskamp, S., Cramer, A. O. J., Waldorp, L. J., Schmittmann, V. D., & Borsboom, D. (2012). qgraph: network visualizations of relationships in psychometric data. *Journal of Statistical Software*, *48*(4), 1–18. Retrieved from http://www.jstatsoft.org/v48/i04/

References II

- Fox, J., Nie, Z., & Byrnes, J. (2013). *sem: structural equation models*. R package version 3.1-1. Retrieved from http://CRAN.R-project.org/package=sem
- Hallquist, M. & Wiley, J. (2013). *MplusAutomation: automating mplus model estimation and interpretation*. R package version 0.5-4. Retrieved from http://CRAN.R-project.org/package=MplusAutomation
- Jöreskog, K. G. & Sörbom, D. (1996). *LISREL 8: user's reference guide*. Scientific Software.
- Muthén, L. K. & Muthén, B. O. (1998–2012). *Mplus user's guide.* (Seventh Edition). Los Angeles, CA: Muthén & Muthén.
- R Core Team. (2013). *R: a language and environment for statistical computing*. ISBN 3-900051-07-0. R Foundation for Statistical Computing. Vienna, Austria. Retrieved from http://www.R-project.org/

References III

- Reingold, E. M. & Tilford, J. S. (1981). Tidier drawings of trees. *Software Engineering, IEEE Transactions on*, *SE-7*(2), 223–228.
- Revelle, W. (2010). PSYCH: procedures for psychological, psychometric, and personality research. R package version 1.0-93. Northwestern University. Evanston, Illinois. Retrieved from http://personality-project.org/r/psych.manual.pdf
- Rosseel, Y. (2012). lavaan: an R package for structural equation modeling. *Journal of Statistical Software*, *48*(2), 1–36. Retrieved from http://www.jstatsoft.org/v48/i02/