lavaan: an R package for structural equation modeling

Yves Rosseel
Department of Data Analysis
Ghent University – Belgium

Modern Modeling Methods Conference
May 22, 2012 – University of Connecticut

Overview

1. software for SEM
2. what is lavaan? why do we need lavaan?
3. features of lavaan
4. lavaan and the (computational) history of SEM

Software for SEM (commercial)

The big four

- LISREL
- EQS
- AMOS
- MPLUS

Others

- CALIS/TCALIS (SAS/Stat)
- SEPATH (Statistica)
- RAMONA (Systat)
- Stata 12
- ...

Software for SEM: non-commercial

- outside the R ecosystem: Mx, gllamm (Stata), ...
- three R packages:
  1. sem
     - developer: John Fox (since 2001)
     - for a long time the only option in R
  2. OpenMx
     - Mx reborn – website: `http://openmx.psyc.virginia.edu/`
     - free, but the solver (NPSOL) is (currently) not open-source
  3. lavaan
     - first public release: May 2010 (version 0.3-1)
     - current version: 0.4-14
- interfaces between R and commercial packages:
  - `REQS`, `MplusAutomation`

What is lavaan?

- **lavaan** is an R package for latent variable analysis:
  - confirmatory factor analysis: function `cfa()`
  - structural equation modeling: function `sem()`
  - latent curve analysis / growth modeling: function `growth()`
  - general mean/covariance structure modeling: function `lavaan()`
  - (multilevel models, latent class + mixture models, Bayesian SEM,...)
- the **lavaan** package is developed to provide applied researchers, teachers, and statisticians a free, open-source, but commercial-quality package for latent variable modeling
- the long-term goal of **lavaan** is
  1. to implement all the state-of-the-art capabilities that are currently available in commercial packages
  2. to provide a modular and extensible platform that allows for easy implementation and testing of new statistical and modeling ideas

Installing lavaan, finding documentation

- **lavaan** depends on the R project for statistical computing:

  `http://www.r-project.org`

- to install **lavaan**, simply start up an R session and type:

```
> install.packages("lavaan")
```

- more information about **lavaan**:

  `http://lavaan.org`

- journal paper:

  Rosseel (2012). lavaan: an R package for structural equation modeling. *Journal of Statistical Software*, 48(2), 1–36.

- **lavaan** development:

  `https://github.com/yrosseel/lavaan`

Why do we need lavaan?

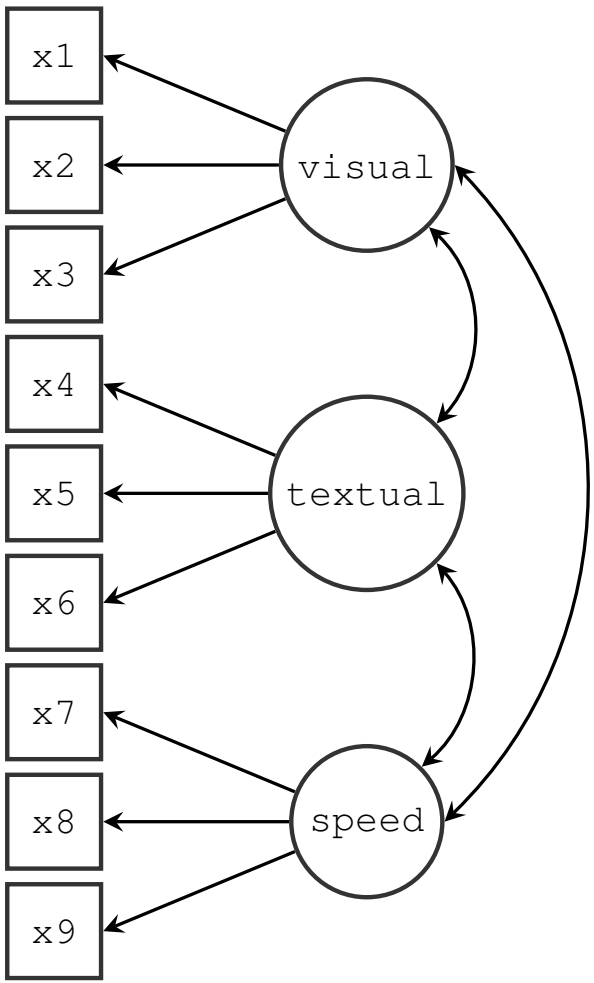
1. lavaan is for statisticians working in the field of SEM
    - it seems unfortunate that new developments in this field are hindered by the lack of open source software that researchers can use to implement their newest ideas
2. lavaan is for teachers
    - teaching these techniques to students was often complicated by the forced choice for one of the commercial packages
3. lavaan is for applied researchers
    - keep it simple, provide all the features they need

Features of lavaan

lavaan is reliable and robust

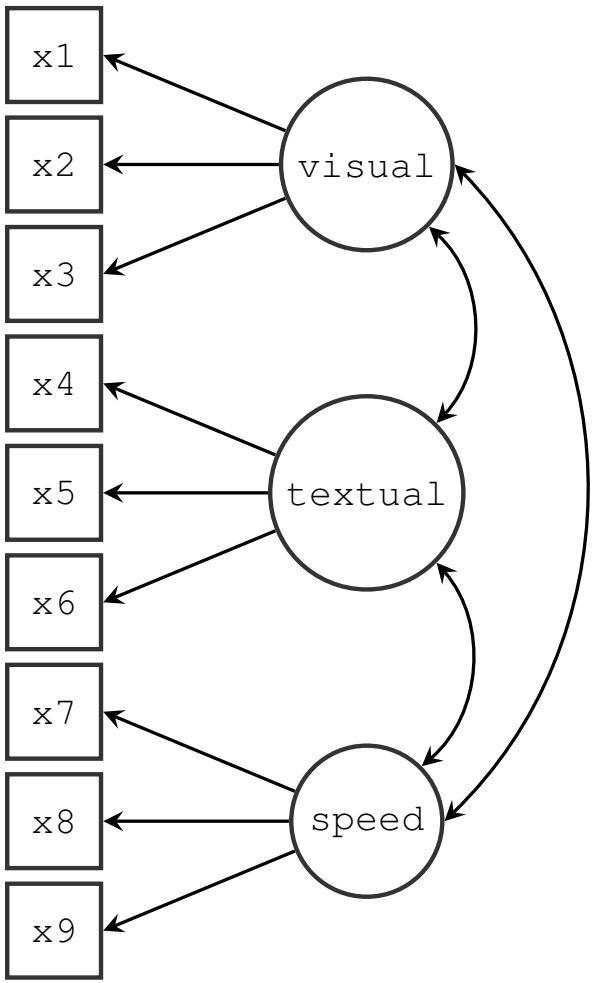
- extensive testing before a 'public' release on CRAN
- no convergence problems (for admissible models)
- numerical results are very close (if not identical) to commercial packages:
  - Mplus (if `mimic="Mplus"`)
  - EQS (if `mimic="EQS"`)

the lavaan model syntax – using cfa() or sem()




```
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9
             '

fit <- cfa(model = HS.model,
           data  = HolzingerSwineford1939,
           estimator = "MLM")

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

the lavaan model syntax – using lavaan()




```
HS.model <- '
  # latent variables
    visual  =~ 1*x1 + x2 + x3
    textual =~ 1*x4 + x5 + x6
    speed   =~ 1*x7 + x8 + x9

  # factor (co)variances
    visual  ~~ visual; visual  ~~ textual
    visual  ~~ speed;  textual ~~ textual
    textual ~~ speed;  speed   ~~ speed

  # residual variances
    x1 ~~ x1; x2 ~~ x2; x3 ~~ x3
    x4 ~~ x4; x5 ~~ x5; x6 ~~ x6
    x7 ~~ x7; x8 ~~ x8; x9 ~~ x9
'

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

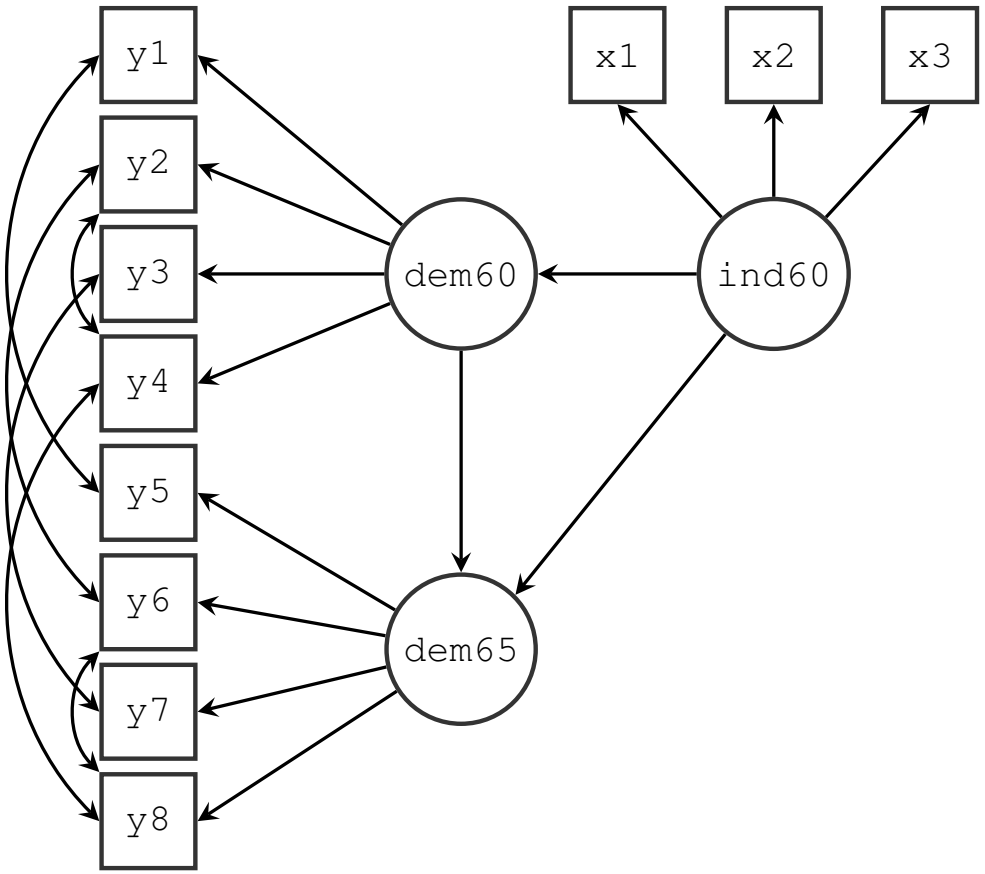
lavaan output

```
lavaan (0.4-14) converged normally after 41 iterations

  Number of observations                           301

  Estimator                                         ML      Robust
  Minimum Function Chi-square                   85.306      80.872
  Degrees of freedom                            24          24
  P-value                                        0.000       0.000
  Scaling correction factor                                  1.055
    for the Satorra-Bentler correction

Chi-square test baseline model:

  Minimum Function Chi-square                  918.852     789.298
  Degrees of freedom                            36          36
  P-value                                        0.000       0.000

Full model versus baseline model:

  Comparative Fit Index (CFI)                    0.931       0.925
  Tucker-Lewis Index (TLI)                       0.896       0.887

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -3737.745   -3737.745
  Loglikelihood unrestricted model (H1)      -3695.092   -3695.092
```

```
  Number of free parameters                         30          30
  Akaike (AIC)                                7535.490    7535.490
  Bayesian (BIC)                              7646.703    7646.703
  Sample-size adjusted Bayesian (BIC)         7551.560    7551.560

Root Mean Square Error of Approximation:

  RMSEA                                          0.092       0.089
  90 Percent Confidence Interval          0.071  0.114       0.068  0.110
  P-value RMSEA <= 0.05                          0.001       0.001

Standardized Root Mean Square Residual:

  SRMR                                           0.060       0.060

Parameter estimates:

  Information                                 Expected
  Standard Errors                             Standard

                   Estimate  Std.err  Z-value  P(>|z|)   Std.lv  Std.all
Latent variables:
  visual =~
    x1                1.000                               0.900    0.772
    x2                0.553    0.100    5.554    0.000    0.498    0.424
    x3                0.729    0.109    6.685    0.000    0.656    0.581
  textual =~
```

```
    x4                1.000                                    0.990    0.852
    x5                1.113    0.065   17.014    0.000         1.102    0.855
    x6                0.926    0.055   16.703    0.000         0.917    0.838
  speed =~
    x7                1.000                                    0.619    0.570
    x8                1.180    0.165    7.152    0.000         0.731    0.723
    x9                1.082    0.151    7.155    0.000         0.670    0.665

Covariances:
  visual ~~
    textual           0.408    0.074    5.552    0.000         0.459    0.459
    speed             0.262    0.056    4.660    0.000         0.471    0.471
  textual ~~
    speed             0.173    0.049    3.518    0.000         0.283    0.283

Variances:
    x1                0.549    0.114                           0.549    0.404
    x2                1.134    0.102                           1.134    0.821
    x3                0.844    0.091                           0.844    0.662
    x4                0.371    0.048                           0.371    0.275
    x5                0.446    0.058                           0.446    0.269
    x6                0.356    0.043                           0.356    0.298
    x7                0.799    0.081                           0.799    0.676
    x8                0.488    0.074                           0.488    0.477
    x9                0.566    0.071                           0.566    0.558
    visual            0.809    0.145                           1.000    1.000
    textual           0.979    0.112                           1.000    1.000
    speed             0.384    0.086                           1.000    1.000
```

lavaan model syntax (2)



lavaan model syntax

```
# latent variable definitions
  ind60 =~ x1 + x2 + x3
  dem60 =~ y1 + y2 + y3 + y4
  dem65 =~ y5 + y6 + y7 + y8

# regressions
  dem60 ~ ind60
  dem65 ~ ind60 + dem60

# residual covariances
  y1 ~~ y5
  y2 ~~ y4
  y2 ~~ y6
  y3 ~~ y7
  y4 ~~ y8
  y6 ~~ y8
```

multiple groups and measurement invariance

```
# model 1: configural invariance
fit1 <- cfa(HS.model, data=HolzingerSwineford1939, group="school")

# model 2: weak invariance
fit2 <- cfa(HS.model, data=HolzingerSwineford1939, group="school",
            group.equal="loadings")

# model 3: strong invariance
fit3 <- cfa(HS.model, data=HolzingerSwineford1939, group="school",
            group.equal=c("loadings", "intercepts"))
```

comparing two (nested) models: the anova() function

```
anova(fit1, fit2)

Chi Square Difference Test

     Df    AIC    BIC  Chisq Chisq diff Df diff Pr(>Chisq)
fit1 48 7484.4 7706.8 115.85
fit2 54 7480.6 7680.8 124.04     8.1922       6     0.2244
```

non-normal and categorical data

- 0.4 version only supports continuous data
- 0.5 version will include support for categorical data
- support for non-normal continuous data
  - asymptotically distribution-free (ADF) estimation (Browne 1984)
  - Satorra-Bentler scaled test statistic and robust standard errors
  - Yuan-Bentler scaled test statistic and robust standard errors when data are both non-normal and missing (at random)
  - bootstrapping: the naïve bootstrap and the Bollen-Stine bootstrap

linear and nonlinear equality and inequality constraints

```
Data <- data.frame(   y = rnorm(100),
                     x1 = rnorm(100),
                     x2 = rnorm(100),
                     x3 = rnorm(100)  )

model.constr <- ' # model with labeled parameters
                    y ~ b1*x1 + b2*x2 + b3*x3

                  # constraints
                    b1 == (b2 + b3)^2
                    b1 > exp(b2 + b3)
                '

fit <- sem(model.constr, data=Data)
```

defined parameters and mediation analysis

```
X <- rnorm(100)
M <- 0.5*X + rnorm(100)
Y <- 0.7*M + rnorm(100)
Data <- data.frame(X = X, Y = Y, M = M)

model <- ' # direct effect
             Y ~ c*X
           # mediator
             M ~ a*X
             Y ~ b*M

           # indirect effect (a*b)
             ab := a*b
           # total effect
             total := c + (a*b)
         '

fit <- sem(model, data=Data)
```

bootstrapping

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

# bootstrapping standard errors
fit <- cfa(HS.model, data=HolzingerSwineford1939, se="bootstrap")

# bootstrapping the test statistic (Bollen-Stine)
fit <- cfa(HS.model, data=HolzingerSwineford1939, test="bootstrap",
           bootstrap=2000, verbose=TRUE)

# bootstrapping anything
fit <- cfa(HS.model, data=HolzingerSwineford1939)

CFI.boot <- bootstrapLavaan(fit, FUN=fitMeasures, R=1000,
                            type="parametric", verbose=TRUE,
                            parallel="multicore", ncups=16,
                            fit.measures="cfi")
```

informative hypothesis testing

- $H_0 : \beta_2 < \beta_1 \quad \text{and} \quad \beta_3 < \beta_1$
- usually done in a Bayesian framework (e.g. Hoijtink, 2012)
- a frequentist approach using parametric bootstrapping in a SEM context has been implemented using a pipeline using R and Mplus (e.g. Van de Schoot R. et.al, 2010)
- Vanbrabant L. et.al (submitted) use the computational infrastructure of lavaan

```
# simple regression model
model  <- 'y1 ~ b1*x1 + b2*x2 + b3*x3'

# constraints
constraints  <- ' b2 < b1
                  b3 < b1 '

InformativeTesting(model = model, data = PoliticalDemocracy,
                   constraints = constraints, R = 10000,
                   double.bootstrap = "FDB")
```

what to expect in lavaan 0.5

- expected release: before the summer (2012)
- support for categorical data (limited information/WLS approach)
- modularity + access to all lavaan internals

what to expect in lavaan 0.6-0.7

- Bayesian estimation
- support for multilevel data
- speed!
- ...
- technical documentation

The history of SEM, from a computational point of view

- several traditions in the SEM (software) world:
  - LISREL (Karl Jöreskog)
  - EQS (Peter Bentler)
  - Mplus (Bengt Muthén)
  - RAM-based approaches (AMOS, Mx, sem, OpenMx, . . .)
- superficially, all SEM software packages produce the same results
- there are some subtle (and less subtle) differences in the output
- looking deeper, there are many computational differences

some differences

- matrix representation
  - standard number of matrices: LISREL: 8; Mplus: 4, EQS: 3, RAM: 2
- optimization algorithm
  - quasi-Newton, gradient-only + quasi-Newton, Gauss-Newton, ...
- variances constrained (strictly positive) versus unrestricted
- constrained optimization algorithm
  - mostly undocumented
  - a Lagrangian-multiplier variant, simple slacks, ...
- normal likelihood versus Wishart likelihood, ML versus GLS-ML (RLS)
  - N versus $N - 1$
  - GLS-ML based chi-square test statistic influences fit measures (CFI!)

some differences (2)

- Satorra-Bentler/Yuan-Bentler scaled test statistic
  - each program seems to use a different implementation
  - often asymptotically equivalent; but large differences in small samples
- categorical data using the limited information approach
  - Muthén 1984; Jöreskog 1994; Lee, Poon, Bentler (1992)
  - many ways to compute the asymptotic covariance matrix (needed for WLS)
- naive bootstrapping, Bollen-Stine bootstrapping
  - mostly undocumented; one-iteration bootstrap?
  - Bollen-Stine with missing data

...

lavaan and the history of SEM

- lavaan is in many areas still trying to catch up with commercial software; but instead of trying to implement one tradition (based on one program), lavaan tries to implement several traditions
- all fitting functions in lavaan have a `mimic` argument which can be set to `"EQS"` or `"Mplus"` respectively; `"LISREL"` is under development
- this was originally intended to convince users that lavaan could produce 'identical' results as the (commercial) competition
- it is now one of the main design goals of lavaan

lavaan and the future of SEM?

- we need to (re)evaluate old/new/unexplored computational methods in many areas (optimization, constrained inference, Bayesian techniques, limited information estimation, ...)
- lavaan should 'by default' implement best practices in all areas

Thank you!

http://lavaan.org