

lava REFERENCE CARD

Linear Latent Variable Models in R

MODEL BUILDING

| | |
|---|---|
| • Initialize model (empty or multivarite regr. model) | <code>m <- lvm(); m <- lvm(c(y1,y2)~x+z)</code> |
| • Initialize from list of regression models | <code>m <- lvm(list(y~x,y~z,...))</code> |
| • Add extra regression associations (slopes) | <code>regression(m) <- c(y1,y3)~u</code> |
| • Add correlation between residual terms | <code>covariance(m) <- y1~y2+y3</code> |
| • Remove associations between variables | <code>cancel(m) <- ~y1+y2</code> |
| • Add variables | <code>addvar(m) <- ~y1+y2</code> |
| • Remove variables | <code>kill(m) <- ~y1+y2</code> |
| • Code as latent (reverse with arg. <code>clear=TRUE</code>) | <code>latent(m) <- ~y1+y2</code> |
| • Binary variables; library(lava.tobit) | <code>binary(m) <- ~y1+y2</code> |

EQUALITY CONSTRAINTS

| | |
|--|---|
| • Intercepts | |
| ◦ Constrain intercepts to be identical | <code>intercept(m) <- c(y1,y2)~f(a)</code> |
| ◦ Simultaneously fix several intercepts | <code>intercept(m,~y1+y2) <- list("a",2)</code> |
| • Variance/covariance parameters | |
| ◦ Fix variance term and covariance between residual to v1 resp. 1 | <code>covariance(m) <- y1~f(y1,v1)+f(y2,1)</code> |
| ◦ Fix multiple variance parameters | <code>covariance(m,~y1+y2) <- list("a",2)</code> |
| ◦ Simultaneously fix several covariance parameters | <code>covariance(m,c(y1,y2)~y2+y3) <- list(2,"a","b",1)</code> |
| • Slope/regression parameters | |
| ◦ $y_1 = x + az + \dots$ $y_2 = x + bx + \dots$ | <code>regression(m,c(y1,y2)~x+z) <- list(1,"a",2,"b")</code> |
| ◦ $y_1 = x + \dots$, $y_2 = az + \dots$ | <code>regression(m,c(y1,y2)~x+z) <- list(1,"a")</code> |
| ◦ $y_i = ax + \dots$ | <code>regression(m) <- c(y1,y2,y3)~f(x,a)</code> |
| • Fix parameters defined by index (see <code>coef</code>) | <code>parfix(m,c(3,4,12)) <- list(1,"a",2)</code> |
| • Label all free parameters (see <code>multigroup</code>) | <code>m <- baptize(m)</code> |
| • Remove linear constraints by fixing to NA (applies also to the <code>intercept</code> and <code>covariance</code> methods) | <code>regression(m) <- c(y1,y2)~f(x,NA)</code> |
| • <i>Bracket notation.</i> Define intercept and variance of residual of y to 0 and 'v' and of x to 'a' and 1. And define $\mathbb{E}(y x) = b \cdot x$. | <code>regression(m) <- y[0:v]~f(x[a,1],b)</code> |

SIMULATION

| | |
|--|--|
| • Simulate 100 observations from model m | <code>sim(m,100,...)</code> |
| • Simulate with the slope-parameter of x on y set to -2 , and intercept of y to 1 (see <code>coef</code>) | <code>sim(m,100,p=c("y"=1, "y<-x"=-2),...)</code> |
| • Define conditional distribution | <code>distribution(m,~y1+y2) <- function(n,mu,var,...) ...</code> |
| ◦ Predefined distributions | <code>binomial.lvm,,uniform.lvm,normal.lvm,poisson.lvm</code> |
| • Define functional form of predictor on response | <code>functional(m,y~x) <- funct</code> |

NON-LINEAR CONSTRAINTS

| | |
|---|--|
| • Non-linear parameter constraints | <code>constrain(m,psi~beta+gamma) <- funct</code> |
| • Non-linear regression (covariate x) | <code>constrain(m,psi~beta+x) <- funct</code> |
| • Add extra parameters to model | <code>parameter(m) <- ~beta+gamma</code> |
| • Add predictor/exogenous variable to model | <code>exogenous(m) <- ~x1+x2</code> |
| • Random slopes (x name of covariate) | <code>regression(m) <- y1~f(eta,x)</code> |
| • Print non-linear constraints | <code>constrain(m)</code> |

MODEL INSPECTION

| | |
|---|--|
| • Examine parameter constraints | <code>intercept,covariance,regression,constrain</code> |
| • Extract variable names | <code>exogenous,endogenous,latent,manifest,vars</code> |
| • Submodel (see also <code>measurement</code>) | <code>subset(m,~y1+y2+eta+x)</code> |
| • List parameter names | <code>coef(m,mean=TRUE,labels=FALSE,...)</code> |
| • Parents and children of nodes (union)) | <code>parents(m,~y1+y2);children(m,~x1+x2)</code> |
| • Extract (directed) pathways between variables | <code>path(m,y~x)</code> |

PLOTTING

| | |
|--|---|
| • Plot method (<code>lvm</code> and <code>lvmfit</code>) | <code>plot(m,labels=TRUE,...)</code> |
| • Change appearance of nodes | <code>nodecolor(m,~y1+x,labcol=c("red","blue"),border,lwd=2,...) <- c("blue","red")</code> |
| • Change label and appearance of edges | <code>edgelabels(m,y~x+z,col,...) <- expression(rho)</code> |
| • Change labels of nodes (e.g. math expressions) | <code>labels(m) <- c(eta=expression(eta))</code> |
| • Extract <code>graphNEL</code> object (library(<code>Rgraphviz</code>)) | <code>Graph(m)</code> |

STATISTICAL INFERENCE

| | |
|--|---|
| • Estimate parameters (default MLE) | <code>e <- estimate(m,data,estimator,...)</code> |
| • Estimate multigroup model (default MLE) | <code>estimate(list(m,...),list(data,...),...)</code> |
| • Estimate under MAR assumption | <code>estimate(m,data,missing=TRUE,...)</code> |
| • Likelihood ratio tests vs. saturated model | <code>compare(e)</code> |
| • Likelihood ratio tests | <code>compare(e1,e2,e3,...)</code> |
| • Model indices based on score tests (or Wald tests) | <code>modelsearch(e,...)</code> |
| • Identify empirical equivalent models | <code>equivalence(e,y~x,k=1,...)</code> |
| • Calculate indirect and total effects of x on y | <code>effects(e, y~x)</code> |
| • Non-linear constraints and approx. std.errors | <code>constraints(e)</code> |
| • Mixtures of LLVM; library(lava.mixture) | <code>mixture(list(m1,m2),data)</code> |
| • Extract various likelihood summaries | <code>coef,score,information,logLik,AIC,gof</code> |
| • Clustered correlated data | <code>estimate(m,data,cluster="id",...)</code> |
| • Robust standard errors | <code>coef(e,type="robust")</code> |
| • Test for linearity; library(<code>gof</code>) | <code>cumres(e,...)</code> |
| • Non-parametric bootstrap | <code>bootstrap(e,R=100,...)</code> |
| • Likelihood-based confidence limits | <code>confint(e,idx,profile=TRUE,...)</code> |

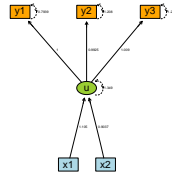
EXAMPLES

Linear Latent Variable Models in R

STRUCTURAL EQUATION MODEL

MIMIC model

```
> m <- lvm(c(y1,y2,y3)~u)
> regression(m) <- u ~ x1+x2
> latent(m) <- ~ u
> d <- sim(m,100)
> e <- estimate(m,d)
> plot(e)
```



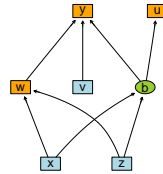
RANDOM REGRESSION

Random slopes allowing for unbalanced designs

```
> m <- lvm(c(y1,y2,y3)~f(eta,1))
> regression(m,c(y1,y2,y3)~u) <- list("x1","x2","x3")
> intercept(m,~y1+y2+y3) <- list("mu")
> covariance(m,~y1+y2+y3) <- list("v","v","v")
> latent(m) <- ~u+eta
> estimate(m,data,missing=TRUE)
```

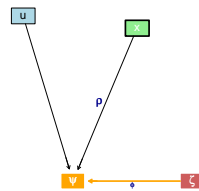
GRAPHICS I

```
> m <- lvm(list(y ~ b+v+w, c(b,w) ~ x+z, u ~ b))
> latent(m) <- ~b
> plot(m)
```



GRAPHICS II

```
> m <- lvm(y~x+z+u)
> labels(m) <- c(y=expression(psi), z=expression(zeta))
> nodecolor(m,~y+z+x,border=c("white","white","black"),
+   labcol="white", lwd=c(0,0,5)) <-
+   c("orange","indianred","lightgreen")
> edgelabels(m,y~z+x, cex=c(2,3), col=c("orange","black"),
+   labcol="darkblue",lwd=c(3,1)) <- expression(phi,rho)
> plot(m,layoutType="circo")
```



INSTRUMENTAL VARIABLE

IV estimator (not available with e.g. non-recursive structures)

```
> estimate(m,data,estimator="IV")
```

SIMULATION

Weibull with exponential distributed censoring

```
> m <- lvm(y~x1+x2+x3)
> distribution(m,~y) <- weibull.lvm(shape=0.5,cens=rexp)
> distribution(m,~x3) <- binomial.lvm()
> d <- sim(m,100)
```

MULTIVARIATE PROBIT

```
> m <- lvm(c(y1,y2)~f(x,b)+f(z,b))
> binary(m) <- ~y1+y2
> covariance(m) <- y1~y2
> estimate(m,data,control=list(trace=1))
```

MULTIGROUP ANALYSIS I

$$\log L(\theta|d) = \sum_i \log L_i(\theta|d_i)$$

```
> estimate(list(m1,m2,m3),list(d1,d2,d3))
```

INDIRECT EFFECTS. TOBIT/PROBIT MODEL

$$\mathbb{E}(y \mid x, z) = a(x + z)$$

```
> m <- lvm(list(y~z+x,z~x))
> d <- transform(sim(m,100),z=factor(z>0),y=Surv(ifelse(y<1,y,1),y<1))
> e <- estimate(m,d)
> effects(e,y~x)
```

NON-LINEAR REGRESSION

Bi-variate non-linear regression with random intercept. Estimated via Fischer scoring

```
> m0 <- lvm(c(y1,y2)~f(x,0)+f(eta,1))
> latent(m0) <- ~eta
> covariance(m0) <- c(y1,y2) ~ f(0.01)
> covariance(m0) <- c(eta) ~ f(0.01)
> d <- sim(m0,50)[,manifest(m0)]
> d <- transform(d,
+   y1=y1+pnorm(2*x),
+   y2=y2+pnorm(2*x))
> m <- m0
> parameter(m) <- ~ nu+alpha+xi
> intercept(m) <- c(y1,y2) ~ f(mu)
> covariance(m) <- c(y1,y2) ~ f(v)
> covariance(m) <- eta ~ f(zeta)
> constrain(m, mu ~ x + alpha + nu + xi) <- function(x) pnorm(x[1]*x[2]+x[4]) + x[3]
> e <- estimate(m,d,control=list(trace=1,method="NR",gamma=0.99))
```