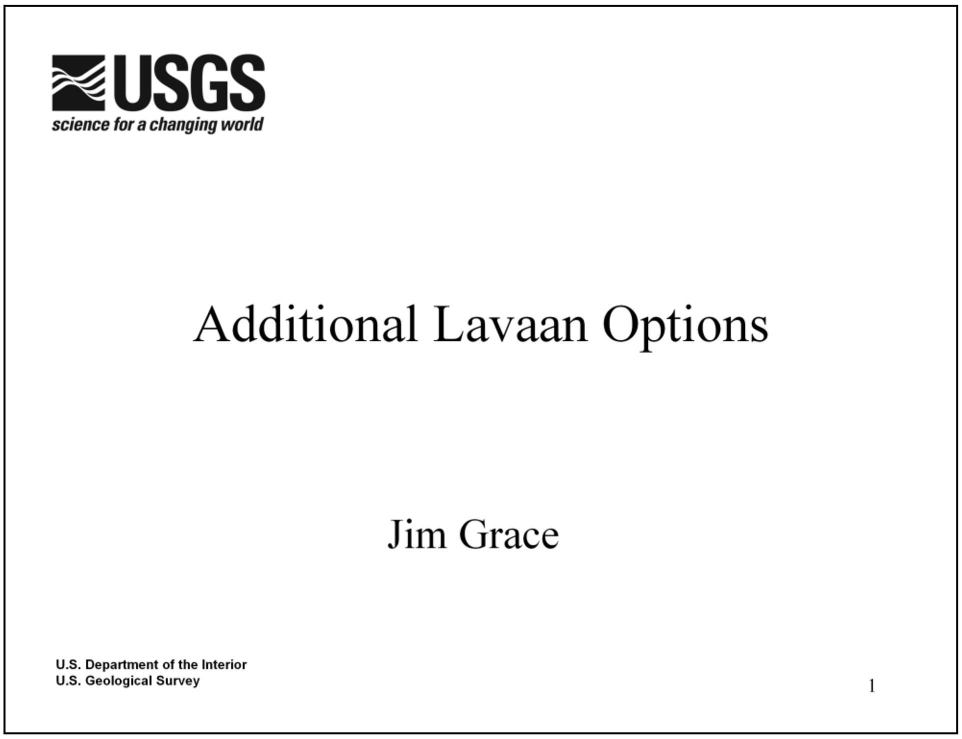


USGS
science for a changing world

Additional Lavaan Options

Jim Grace

U.S. Department of the Interior
U.S. Geological Survey

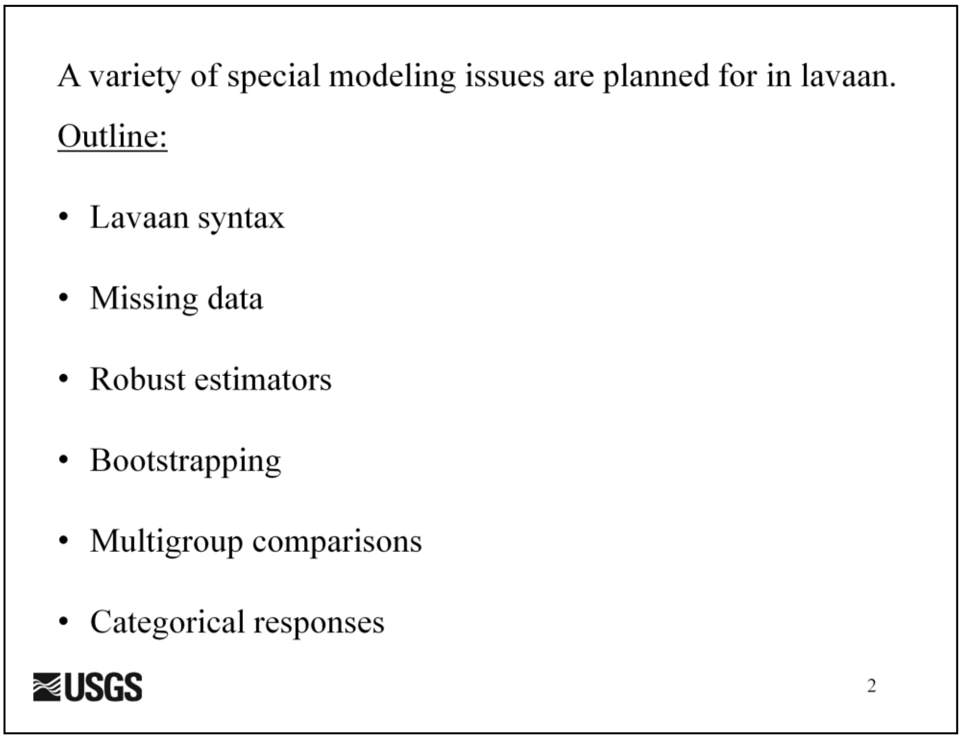
1

In this module I provide a few illustrations of options within lavaan for handling various situations.

An appropriate citation for this material is

Yves Rosseel (2012). lavaan: An R Package for Structural Equation Modeling. Journal of Statistical Software, 48(2), 1-36. URL http://www.jstatsoft.org/v48/i02/

Notes: IP-056512; Support provided by the USGS Climate & Land Use R&D and Ecosystems Programs. I would like to acknowledge formal review of this material by Jesse Miller and Phil Hahn, University of Wisconsin. Many helpful informal comments have contributed to the final version of this presentation. The use of trade names is for descriptive purposes only and does not imply endorsement by the U.S. Government. Last revised 20141216. Questions about this material can be sent to sem@usgs.gov.

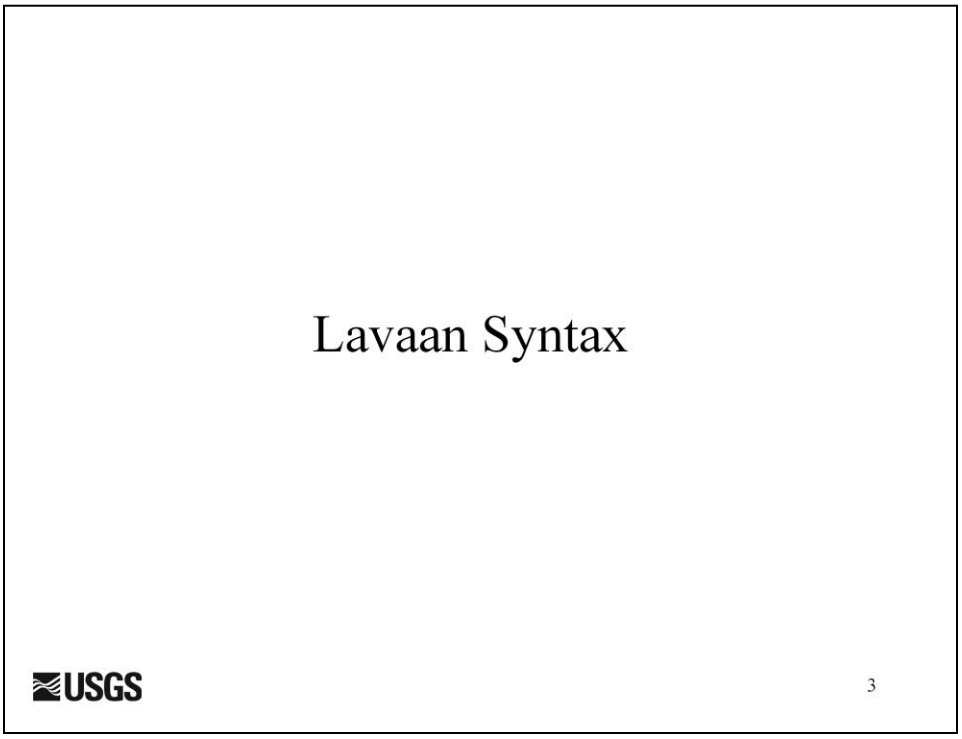


A variety of special modeling issues are planned for in lavaan.

Outline:

- Lavaan syntax
- Missing data
- Robust estimators
- Bootstrapping
- Multigroup comparisons
- Categorical responses

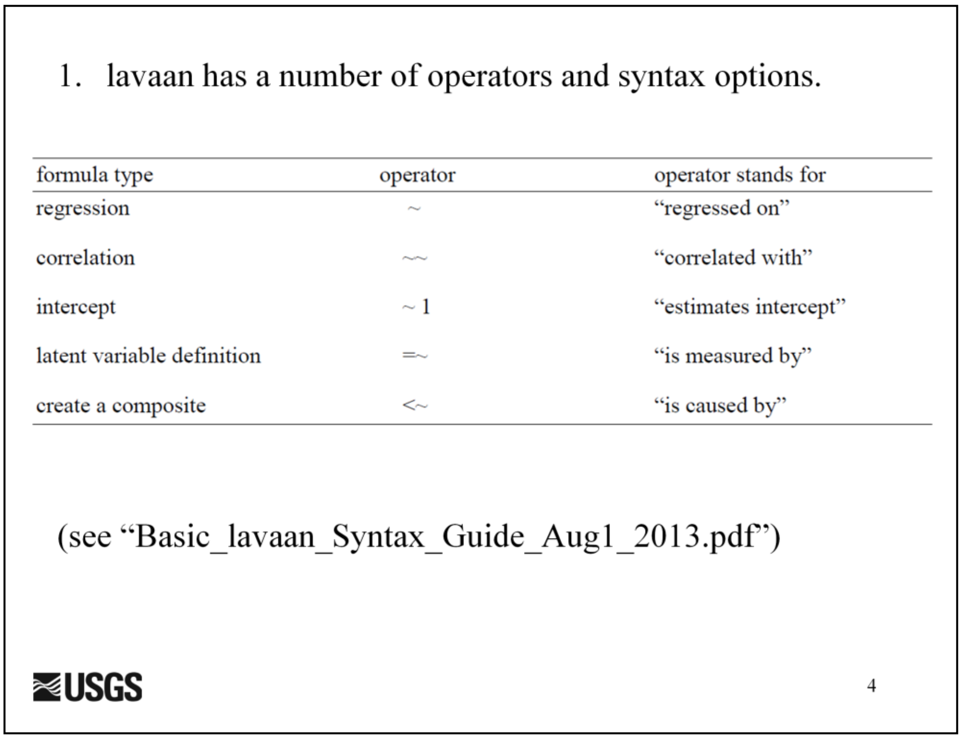
I start with just presenting a table of lavaan syntax. Then I consider four special topics.



Lavaan Syntax

3

Here we revisit the issue of syntax available in lavaan.



1. lavaan has a number of operators and syntax options.

formula type	operator	operator stands for
regression	~	"regressed on"
correlation	~~	"correlated with"
intercept	~ 1	"estimates intercept"
latent variable definition	=~	"is measured by"
create a composite	<~	"is caused by"

(see "Basic_lavaan_Syntax_Guide_Aug1_2013.pdf")

USGS

4

Yves Rosseel's latest (authoratative) tutorial is at:http://lavaan.ugent.be/tutorial/tutorial.pdf

2. You can work with individual parameters by naming them.

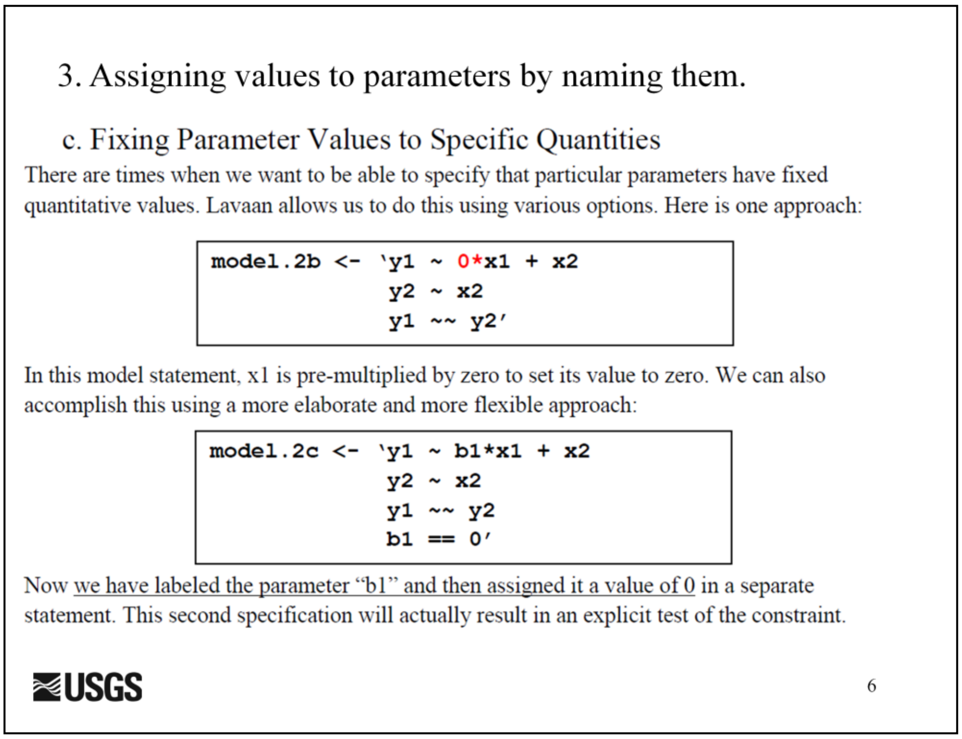
Lavaan names parameters as **“y1 ~ x1”**.

We assign names by pre-multiplying a predictor with the name being assigned.

```
model.2a <- 'y1 ~ b1*x1
```

Note, parameter labels must start with a letter!

Naming parameters is a key step in many operations. Note that the “b1” in the R code names a parameter, which permits us to manually set a parameter value.



3. Assigning values to parameters by naming them.

c. Fixing Parameter Values to Specific Quantities

There are times when we want to be able to specify that particular parameters have fixed quantitative values. Lavaan allows us to do this using various options. Here is one approach:

```
model.2b <- 'y1 ~ 0*x1 + x2
             y2 ~ x2
             y1 ~~ y2'
```

In this model statement, x1 is pre-multiplied by zero to set its value to zero. We can also accomplish this using a more elaborate and more flexible approach:

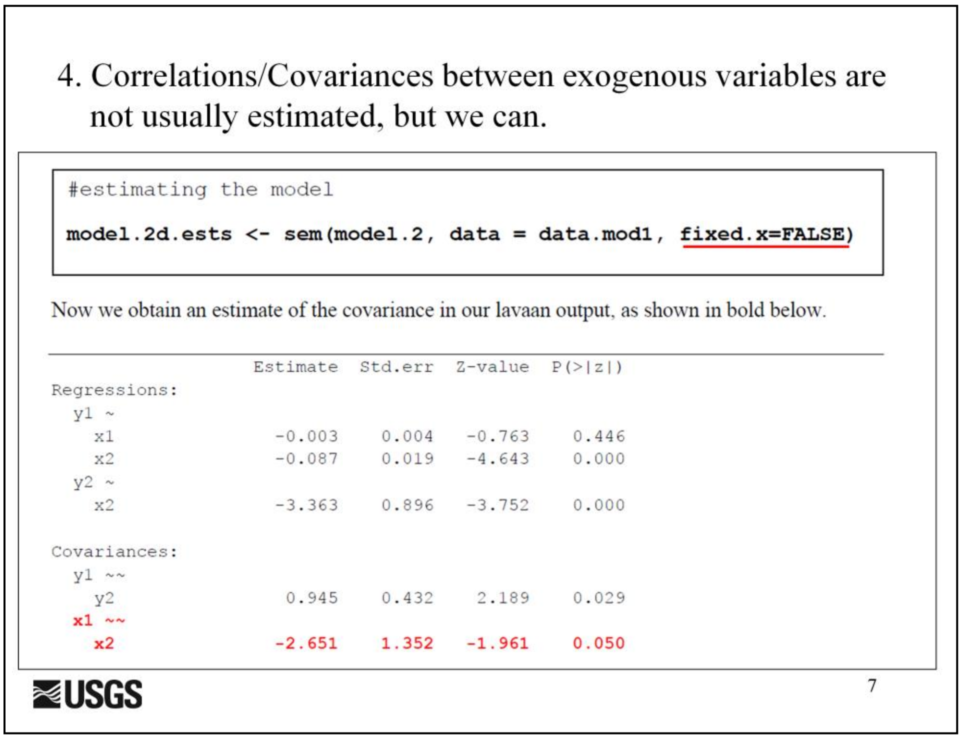
```
model.2c <- 'y1 ~ b1*x1 + x2
             y2 ~ x2
             y1 ~~ y2
             b1 == 0'
```

Now we have labeled the parameter "b1" and then assigned it a value of 0 in a separate statement. This second specification will actually result in an explicit test of the constraint.

USGS

6

Assigning values is also important.



4. Correlations/Covariances between exogenous variables are not usually estimated, but we can.

```
#estimating the model

model.2d.ests <- sem(model.2, data = data.mod1, fixed.x=FALSE)
```

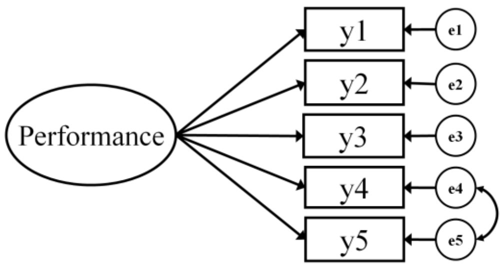
Now we obtain an estimate of the covariance in our lavaan output, as shown in bold below.

```
                      Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  y1 ~
    x1                  -0.003    0.004   -0.763    0.446
    x2                  -0.087    0.019   -4.643    0.000
  y2 ~
    x2                  -3.363    0.896   -3.752    0.000

Covariances:
  y1 ~~
    y2                   0.945    0.432    2.189    0.029
  x1 ~~
    x2                  -2.651    1.352   -1.961    0.050
```

USGS 7

A default of lavaan, like all software except Amos, is the just take the exogenous correlations/covariances directly from the data and not treat them as estimated parameters. The module on “SEM Essentials – Path Rules” explains how that is possible. Anyway, sometime we want or need to treat those as estimated parameters, so the command above shows how.

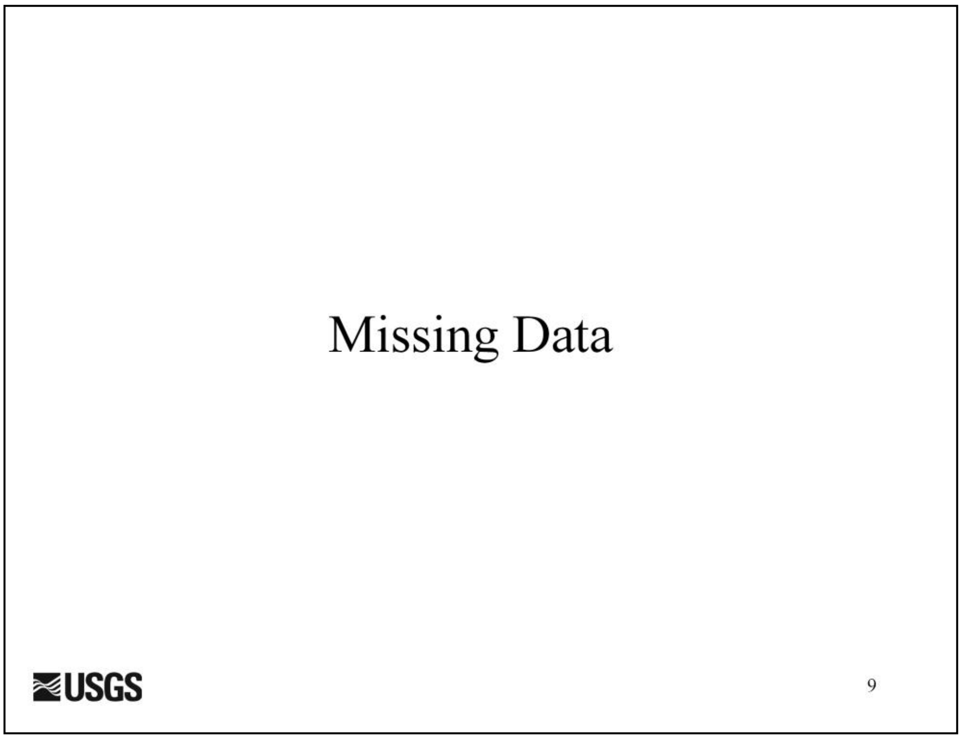
5. Lavaan creates latent variables by declaring them in the absence of any known values.



```
lvmod.2 <- ' # Latent variable definition
              Perform =~ stems + infls + clonediam
                         + leafht + leafwdth

             # Error Covariances
              leafht ~~ leafwdth'
```

8

Covered in greater depth in the module on latent variable modeling.

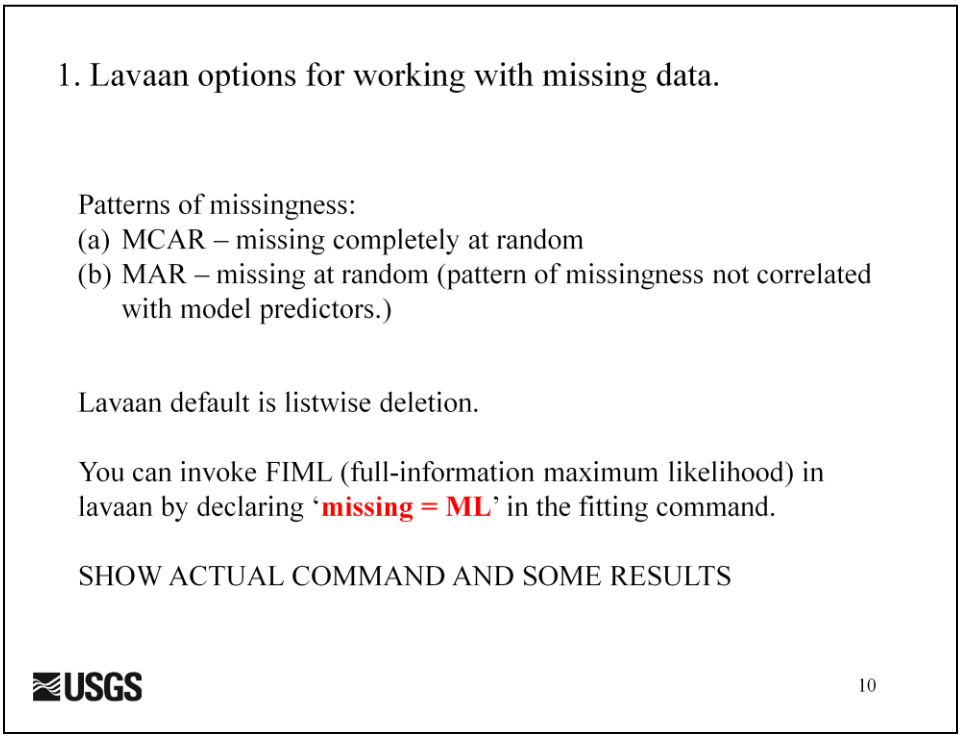


Missing Data

USGS

9

Important issue – dealing with missing data.



1. Lavaan options for working with missing data.

Patterns of missingness:
(a) MCAR – missing completely at random
(b) MAR – missing at random (pattern of missingness not correlated with model predictors.)

Lavaan default is listwise deletion.

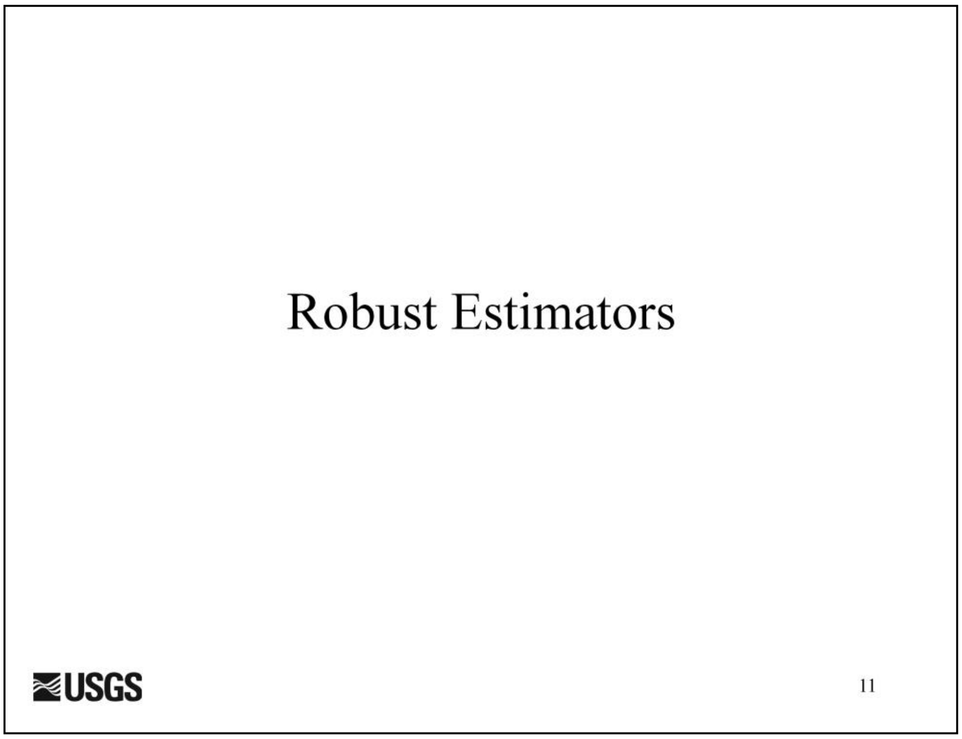
You can invoke FIML (full-information maximum likelihood) in lavaan by declaring '**missing = ML**' in the fitting command.

SHOW ACTUAL COMMAND AND SOME RESULTS

USGS

10

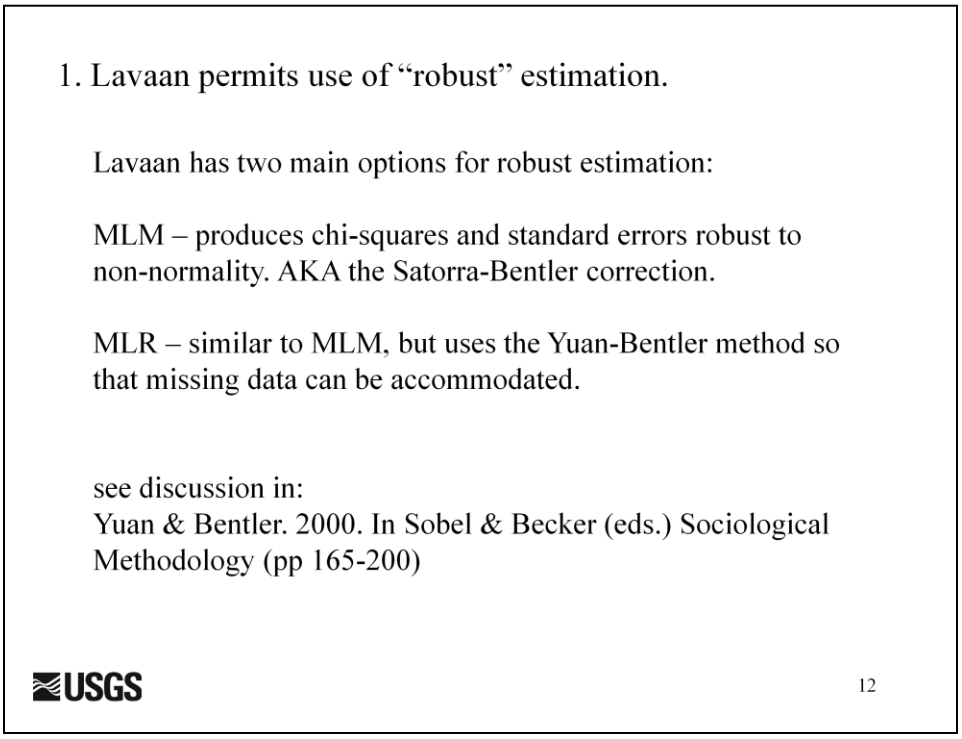
Important topic that I will not cover here, except to let you know that lavaan has a very powerful option for performing analyses in the presence of missing data.



Robust Estimators

11

Methods have been developed to provide estimates that are robust to deviations from the assumption of normal errors. Here we see what lavaan has to offer in that area.



1. Lavaan permits use of "robust" estimation.

Lavaan has two main options for robust estimation:

MLM – produces chi-squares and standard errors robust to non-normality. AKA the Satorra-Bentler correction.

MLR – similar to MLM, but uses the Yuan-Bentler method so that missing data can be accommodated.

see discussion in:
Yuan & Bentler. 2000. In Sobel & Becker (eds.) Sociological Methodology (pp 165-200)

USGS

12

Robust means the inferences are robust to deviations from normality in the response variables.

2. Robust estimation invoked with 'estimator =' command.

```
# create model
mod <- 'y2 ~ y1
        y1 ~ x1'

# estimate model
mod.fit <- sem(mod, data=dat, fixed.x=F,
               estimator="mlm")

# get results
summary(mod.fit)
```

"fixed.x=F" is required when using "mlm" option.

USGS

Declaring the estimator when fitting a lavaan model is simple.

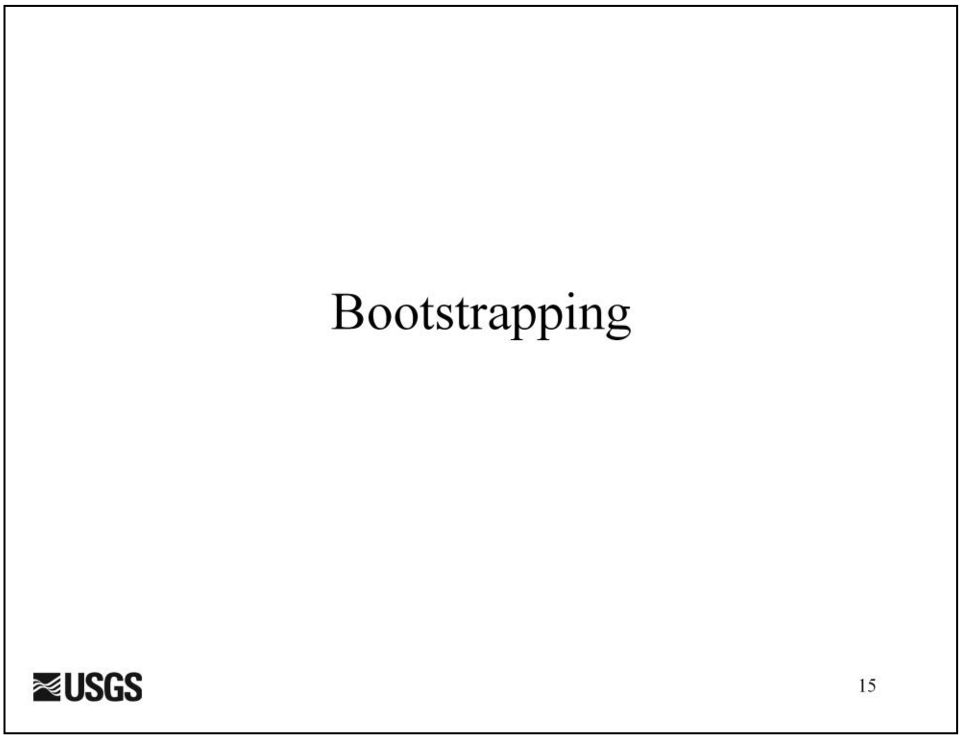
3. Results

```
 Estimator                               ML       Robust
  Chi-square                          4.213        4.082
  Degrees of freedom                      1            1
  P-value                             0.040        0.043
  Scaling correction factor
  for the Yuan-Bentler correction                  1.032
```

```
 Standard Errors                           Robust.mlm

                 Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  y2 ~
    y1            -0.154    0.024   -6.386    0.000
  y1 ~
    x1             1.185    0.290    4.091    0.000
```

USGS

The results output shows the adjusted values.



Bootstrapping

USGS

15

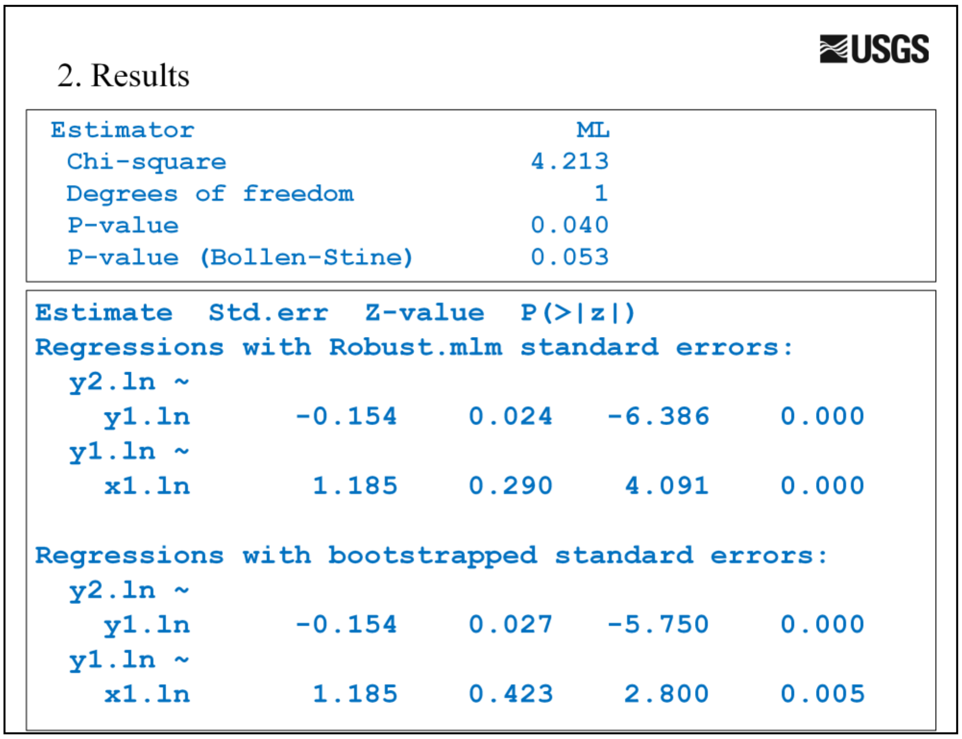
A commonly used approach to estimating probabilities is resampling and one particularly popular form of resampling is bootstrapping (sampling with replacement).

1. Lavaan has resampling methods for non-normal data.

```
# fit model and request bootstrapped results

mod.fit <- sem(mod, dat=dat,
           test="boot", se="boot", bootstrap=200)
```

USGS

Bootstrapping is likewise a simple operation in lavaan.

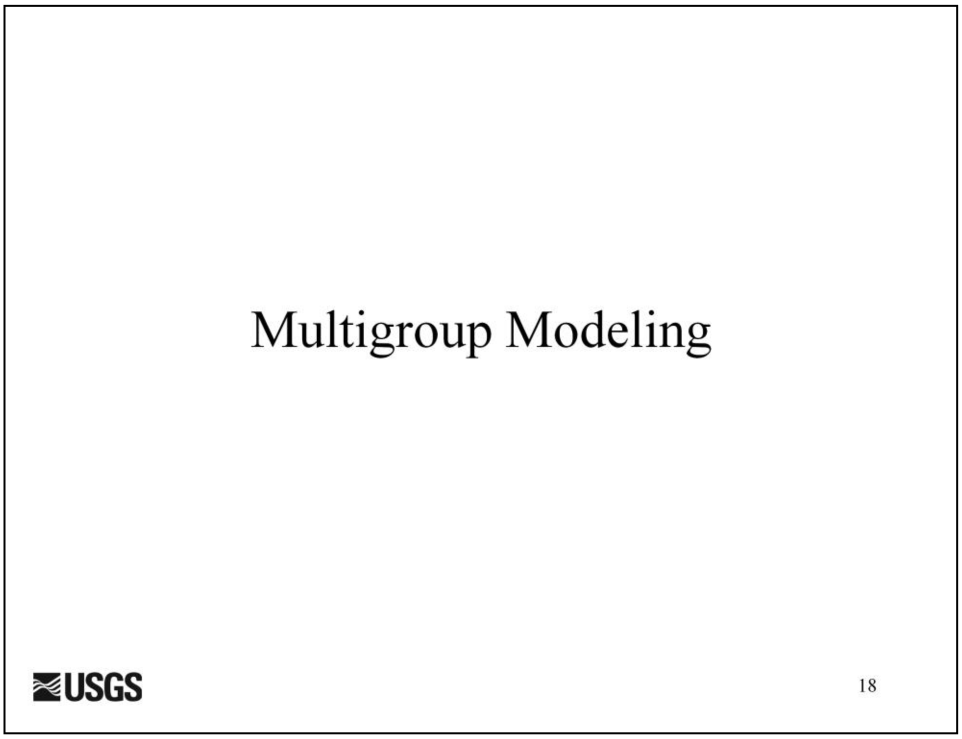


2. Results

```
Estimator                              ML
  Chi-square                        4.213
  Degrees of freedom                    1
  P-value                           0.040
  P-value (Bollen-Stine)            0.053
```

```
Estimate  Std.err  Z-value  P(>|z|)
Regressions with Robust.mlm standard errors:
  y2.ln ~
    y1.ln      -0.154    0.024   -6.386    0.000
  y1.ln ~
    x1.ln       1.185    0.290    4.091    0.000

Regressions with bootstrapped standard errors:
  y2.ln ~
    y1.ln      -0.154    0.027   -5.750    0.000
  y1.ln ~
    x1.ln       1.185    0.423    2.800    0.005
```

We can expect bootstrapped results to give different standard errors and p-values.

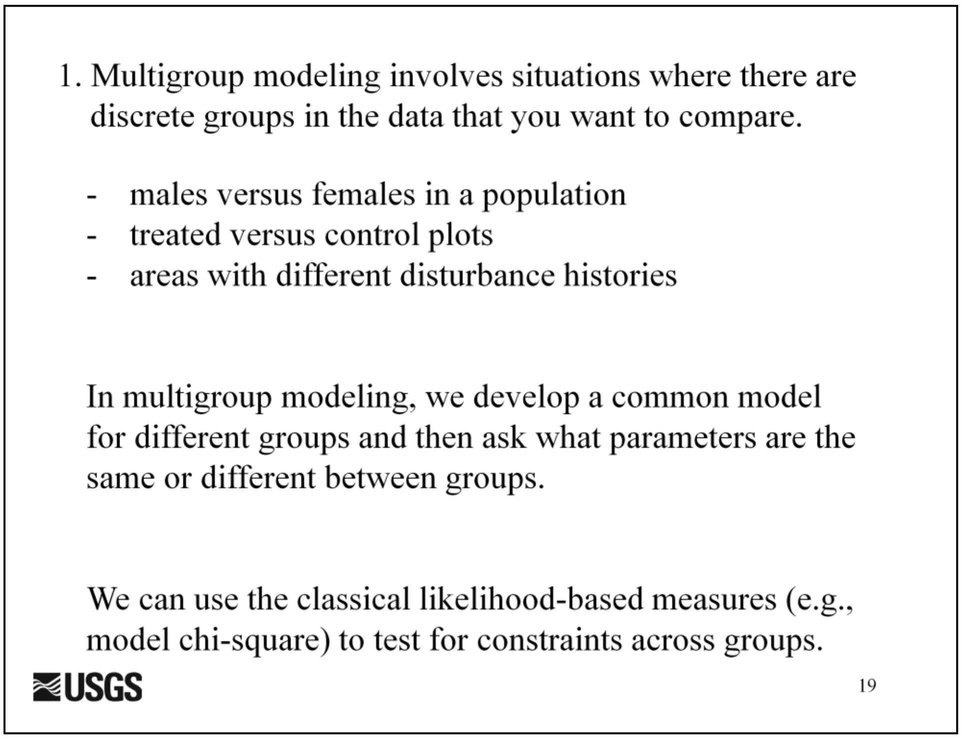


Multigroup Modeling

USGS

18

It is possible to ask whether a common model applies to multiple groups. This is automated in lavaan.



1. Multigroup modeling involves situations where there are discrete groups in the data that you want to compare.

   - males versus females in a population
   - treated versus control plots
   - areas with different disturbance histories

In multigroup modeling, we develop a common model for different groups and then ask what parameters are the same or different between groups.

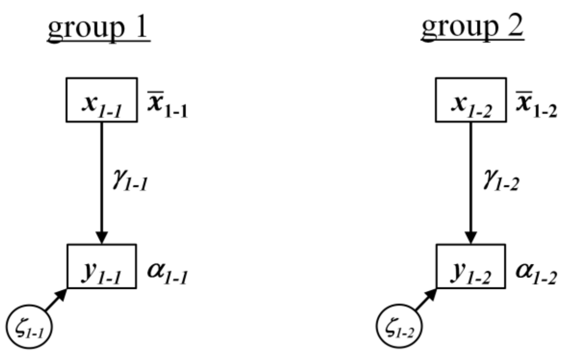
We can use the classical likelihood-based measures (e.g., model chi-square) to test for constraints across groups.

USGS

19

The ability to formally compare groups is very important.

2. There are several parameters that can be compared.



A number of hypotheses we can test:

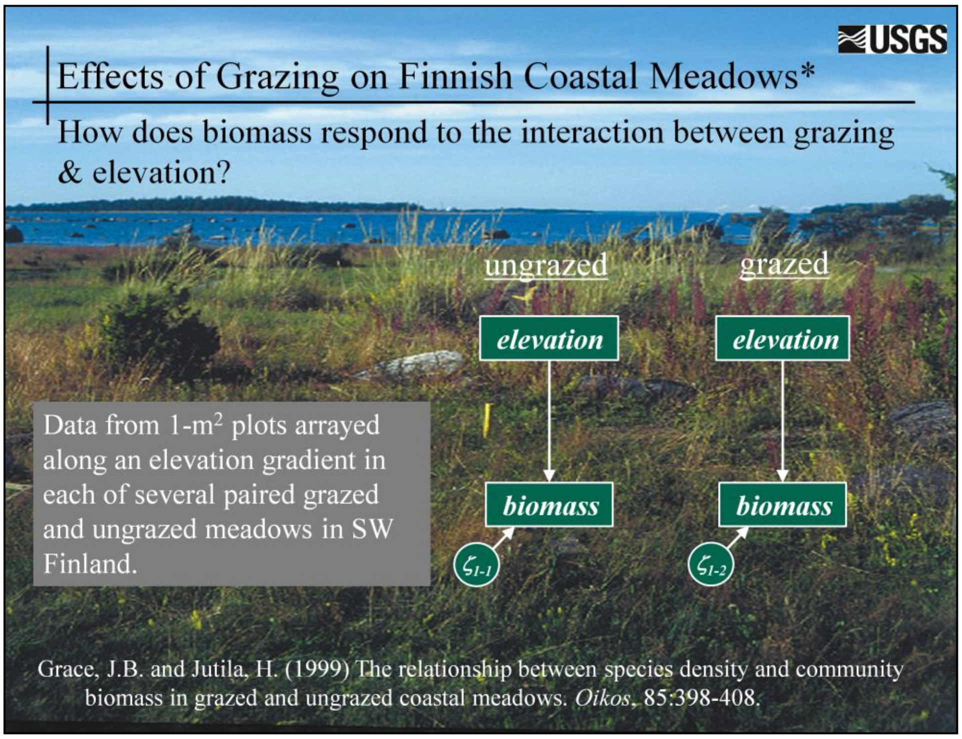
(1) equal slopes:	$\gamma_{1\text{-}1} = \gamma_{1\text{-}2}$
(2) equal intercepts:	$\alpha_{1\text{-}1} = \alpha_{1\text{-}2}$
(3) equal means:	$\bar{x}_{1\text{-}1} = \bar{x}_{1\text{-}2}$
(4) equal errors:	$\zeta_{1\text{-}1} = \zeta_{1\text{-}2}$

$\downarrow$ increasingly strong constraints

20

USGS

We are proposing an overall model that applies to both groups and then testing to see if the raw parameter estimates are essentially the same across groups (meaning a process is common to both groups).



Effects of Grazing on Finnish Coastal Meadows*

How does biomass respond to the interaction between grazing & elevation?

Data from 1-m^2 plots arrayed along an elevation gradient in each of several paired grazed and ungrazed meadows in SW Finland.

ungrazed

$$\begin{tikzcd} \text{elevation} \arrow[d] & \\ \text{biomass} & \zeta_{1-1} \arrow[l] \end{tikzcd}$$

grazed

$$\begin{tikzcd} \text{elevation} \arrow[d] & \\ \text{biomass} & \zeta_{1-2} \arrow[l] \end{tikzcd}$$

Grace, J.B. and Jutila, H. (1999) The relationship between species density and community biomass in grazed and ungrazed coastal meadows. *Oikos*, 85:398-408.

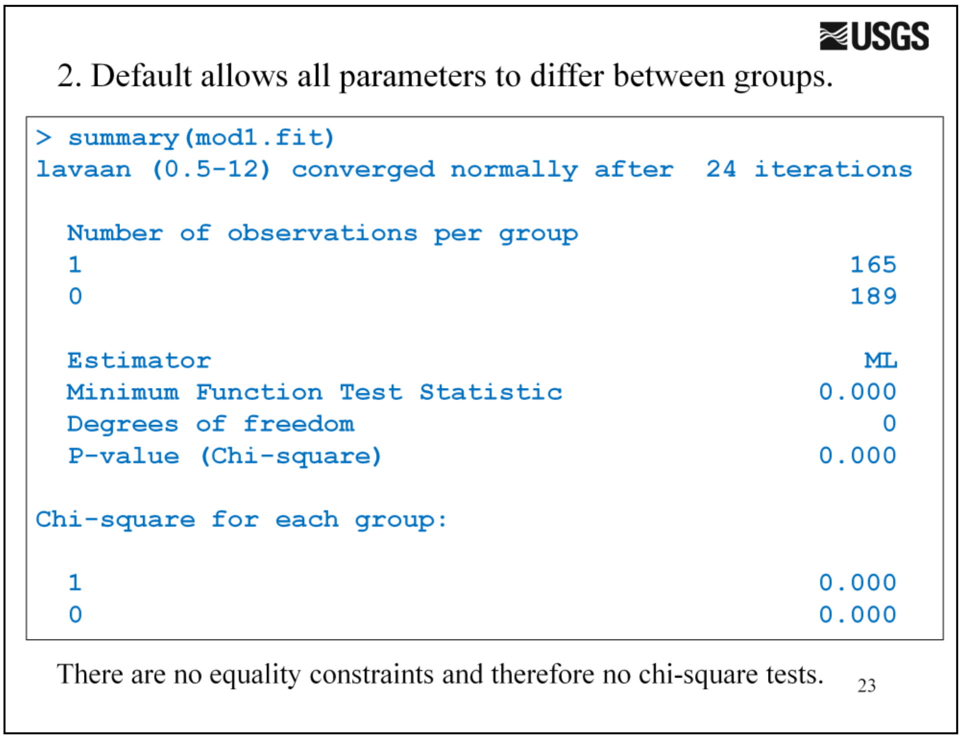
Here is an example.

1. Lavaan uses a ‘group=’ command to invoke a multigroup analysis.

```
#lavaan code for basic model
mod1 <-'biomass ~ elev'

#fit the model, specifying groups
mod1.fit <- sem(mod1, data=sem.dat, group="grazed")

#request output
summary(mod1.fit)
```

USGS

Use “group=“ command to invoke a multi-group modeling setup. Note that here the grouping variable “grazed” is a 0/1 dummy variable with 1==grazed.

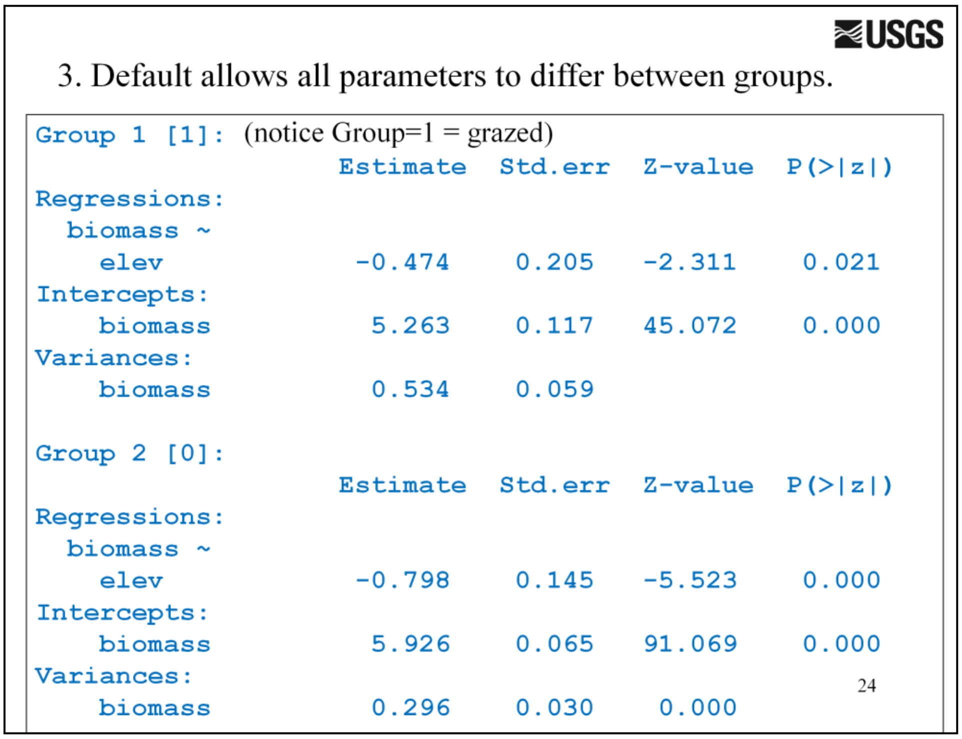


2. Default allows all parameters to differ between groups.

```
> summary(mod1.fit)
lavaan (0.5-12) converged normally after  24 iterations

  Number of observations per group
  1                                                 165
  0                                                 189

  Estimator                                          ML
  Minimum Function Test Statistic                 0.000
  Degrees of freedom                                  0
  P-value (Chi-square)                            0.000

Chi-square for each group:

  1                                               0.000
  0                                               0.000
```

There are no equality constraints and therefore no chi-square tests. 23

Default is to permit all parameters to be unique across groups. Notice the number of observations for the groups is given.



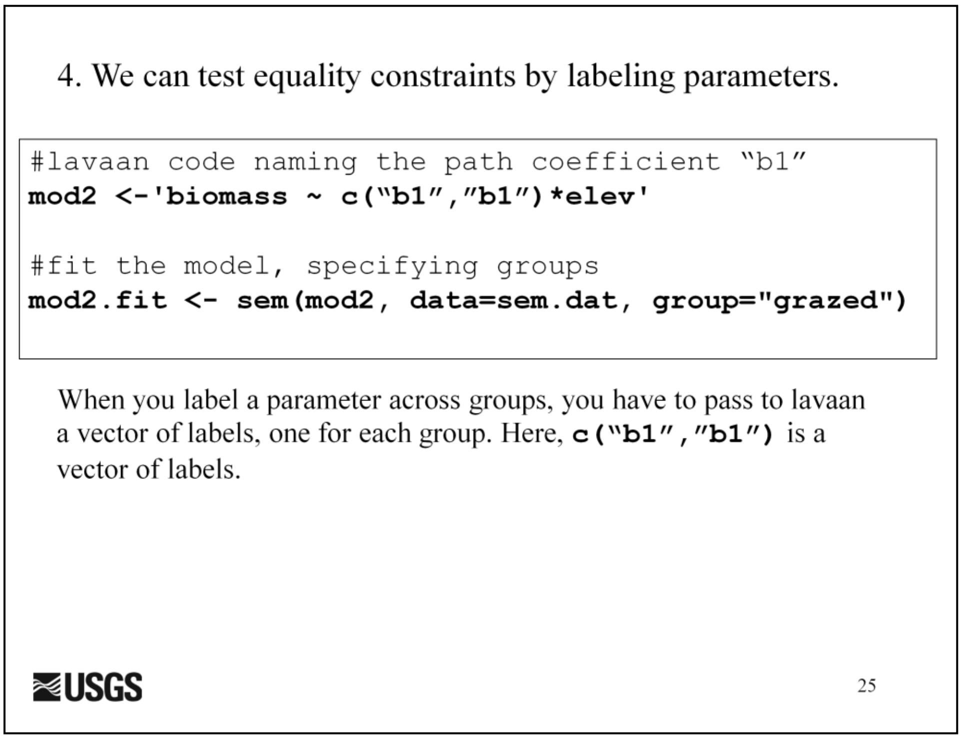
3. Default allows all parameters to differ between groups.

Group 1 [1]: (notice Group=1 = grazed)

```
                     Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  biomass ~
    elev               -0.474    0.205   -2.311    0.021
Intercepts:
    biomass             5.263    0.117   45.072    0.000
Variances:
    biomass             0.534    0.059

Group 2 [0]:
                     Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  biomass ~
    elev               -0.798    0.145   -5.523    0.000
Intercepts:
    biomass             5.926    0.065   91.069    0.000
Variances:
    biomass             0.296    0.030    0.000
```

24

Group results are presented separately.



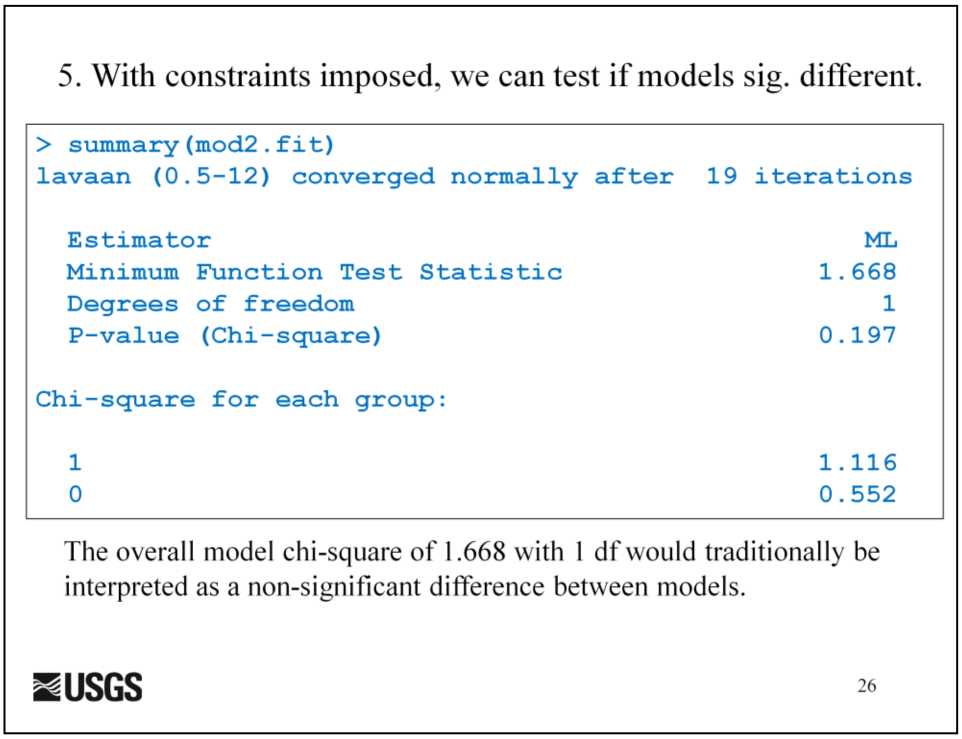
4. We can test equality constraints by labeling parameters.

```
#lavaan code naming the path coefficient “b1”
mod2 <-'biomass ~ c(“b1”,”b1”)*elev'

#fit the model, specifying groups
mod2.fit <- sem(mod2, data=sem.dat, group="grazed")
```

When you label a parameter across groups, you have to pass to lavaan a vector of labels, one for each group. Here, **`c(“b1”,”b1”)`** is a vector of labels.

25

So, our initial analysis allows all parameters to be different between groups. We then might like to add constraints sequentially to determine what is the same across groups. There are some general commands in lavaan for this task, but let’s start with a simple general approach – setting a single parameter equal across groups.



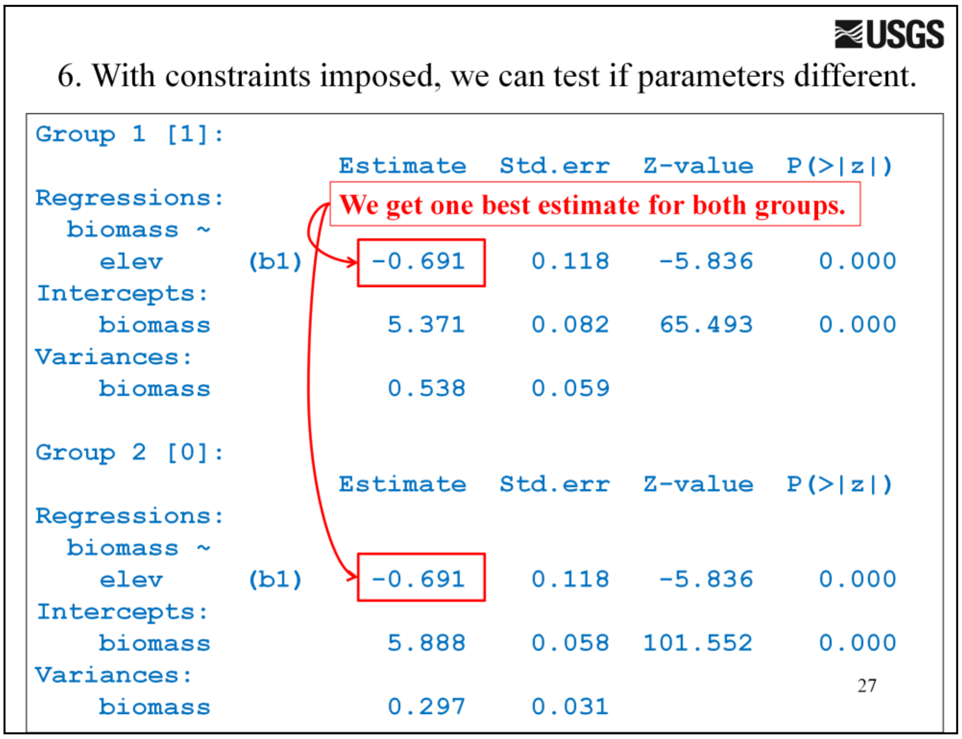
5. With constraints imposed, we can test if models sig. different.

```
> summary(mod2.fit)
lavaan (0.5-12) converged normally after  19 iterations

  Estimator                                         ML
  Minimum Function Test Statistic                1.668
  Degrees of freedom                                 1
  P-value (Chi-square)                           0.197

Chi-square for each group:

  1                                              1.116
  0                                              0.552
```

The overall model chi-square of 1.668 with 1 df would traditionally be interpreted as a non-significant difference between models.

26

Equality constraints reduce the number of parameters being estimated and provide model degrees of freedom for hypothesis tests.



6. With constraints imposed, we can test if parameters different.

We get one best estimate for both groups.

```
Group 1 [1]:
                         Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  biomass ~
    elev        (b1)       -0.691    0.118   -5.836    0.000
Intercepts:
    biomass                 5.371    0.082   65.493    0.000
Variances:
    biomass                 0.538    0.059

Group 2 [0]:
                         Estimate  Std.err  Z-value  P(>|z|)
Regressions:
  biomass ~
    elev        (b1)       -0.691    0.118   -5.836    0.000
Intercepts:
    biomass                 5.888    0.058  101.552    0.000
Variances:
    biomass                 0.297    0.031
```

27

Here we see what is going on.

7. We can also use an 'equal' or a 'group.equal' command.

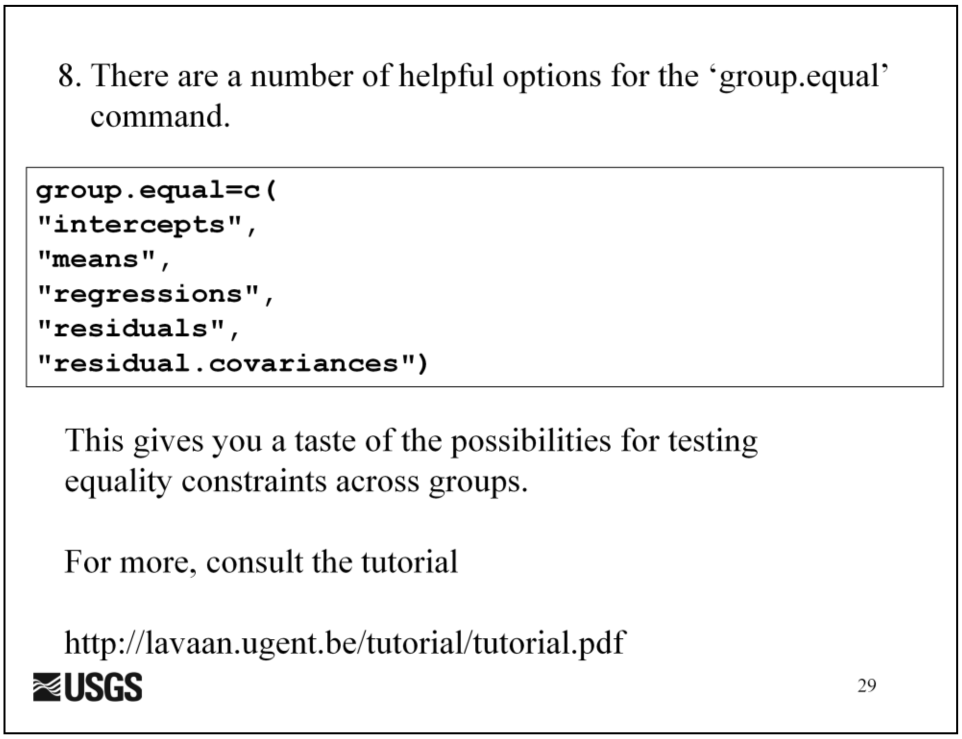
```
mod2a <-'biom.log ~ equal("b1")*elev.m'

mod2a.fit <- sem(mod2a, data=sem.dat, group="grazed")
```

Produces exactly the same results as the previous command.

```
mod3 <-'biom.log ~ elev.m'

mod3.fit <- sem(mod3, data=sem.dat, group="grazed",
group.equal="regressions")
```

USGS

Lavann has other options.



8. There are a number of helpful options for the ‘group.equal’ command.

```
group.equal=c(
"intercepts",
"means",
"regressions",
"residuals",
"residual.covariances")
```

This gives you a taste of the possibilities for testing equality constraints across groups.

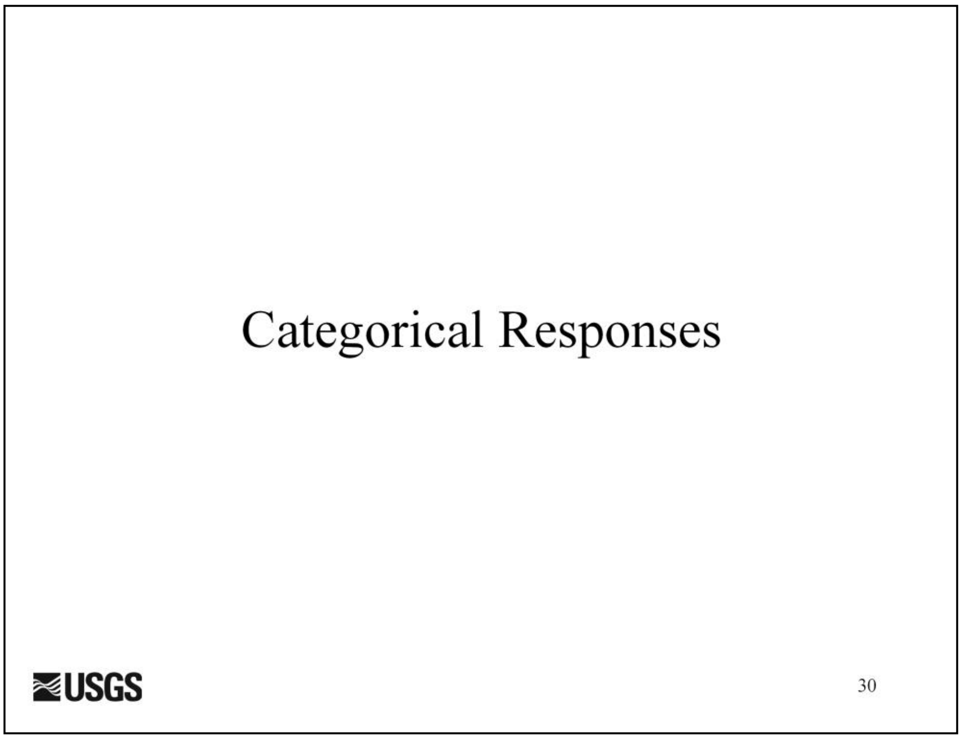
For more, consult the tutorial

http://lavaan.ugent.be/tutorial/tutorial.pdf

USGS

29

Lavaan makes this as automated as possible.



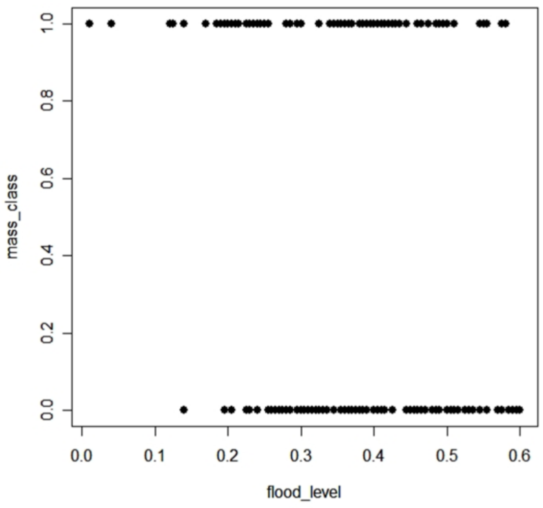
Categorical Responses

≋USGS

30

Another common issue is when one has response variables that are ordered categorical.

The problem of analyzing categorical responses



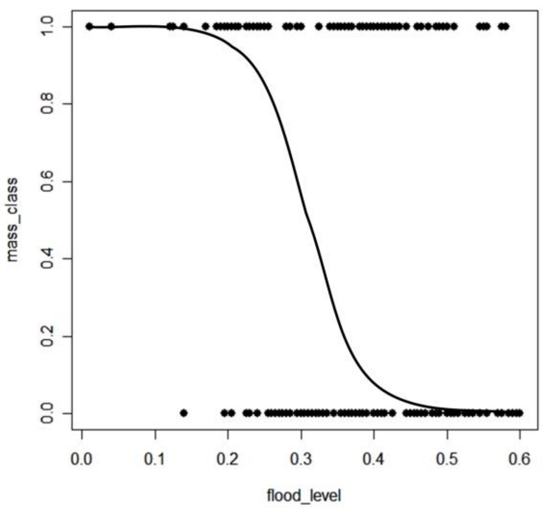
Continuous predictor (flood-level) and binary response (mass class = 0/1)

USGS

31

Fitting a straight line through such a set of points represents the points quite poorly and leads to illogical extrapolations, like intercepts > 1 or < 0. It also violates assumptions about normality of residuals. What we need is a way to interpret binary outcomes that makes sense. Often this is accomplished by assuming that behind the binary outcomes lies a continuous probability of observing a 1 or 0 response, as shown on the next slide.

A model for binary responses



probit and **logit** models are common response models.

Probit model:
link predictor to responses using cumulative normal probability function.

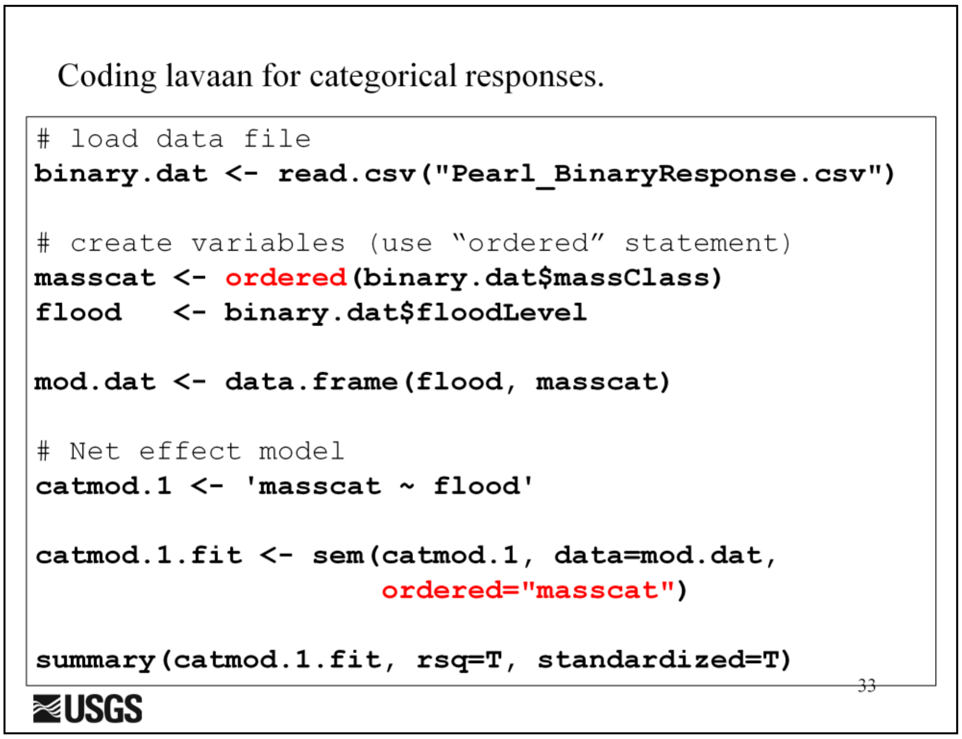
Logit model:
link predictor to responses using log transformation of ratio of probabilities of outcomes.

USGS

32

Two of the most common ways of representing the probability of observing a 1 or 0 outcome are the probit and logit models.

For the probit model, we link our predictor to our responses using a cumulative normal probability function, as shown to the left. With the logit model, we link our predictor to our responses using an inverse log transformation of the ratio of probabilities of outcomes.



Coding lavaan for categorical responses.

```
# load data file
binary.dat <- read.csv("Pearl_BinaryResponse.csv")

# create variables (use "ordered" statement)
masscat <- ordered(binary.dat$massClass)
flood   <- binary.dat$floodLevel

mod.dat <- data.frame(flood, masscat)

# Net effect model
catmod.1 <- 'masscat ~ flood'

catmod.1.fit <- sem(catmod.1, data=mod.dat,
                    ordered="masscat")

summary(catmod.1.fit, rsq=T, standardized=T)
```

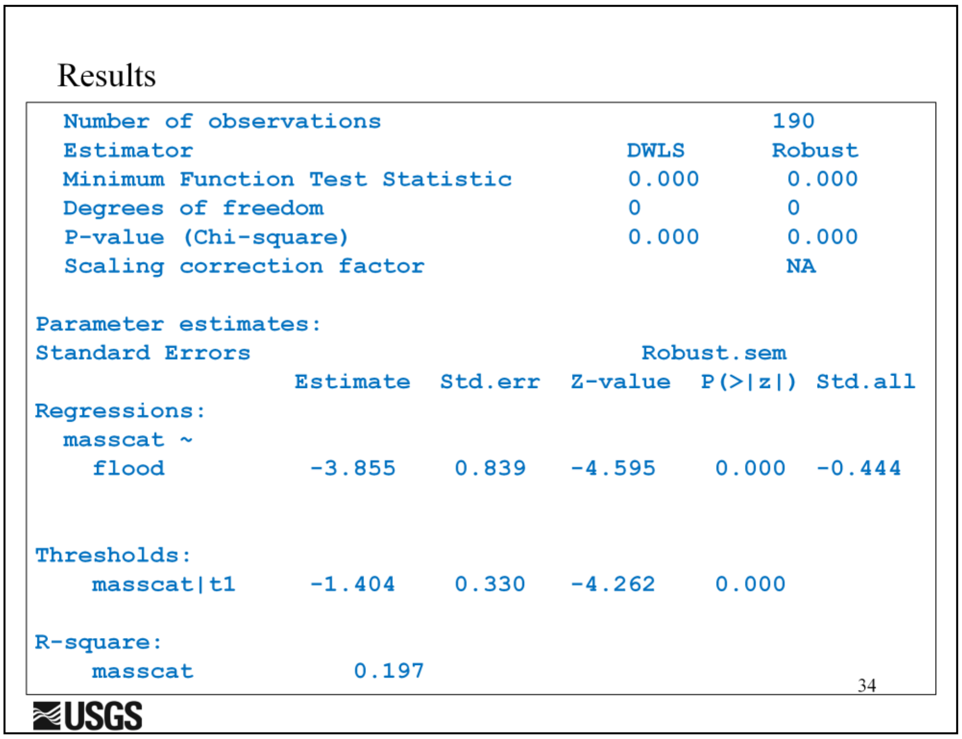
33

USGS

Two requirements

(1) Declare categorical variable as "ordered" object.

(2) Declare variables that are ordered categorical in the "sem" statement.



Results

```
  Number of observations                          190
  Estimator                             DWLS      Robust
  Minimum Function Test Statistic       0.000      0.000
  Degrees of freedom                    0          0
  P-value (Chi-square)                  0.000      0.000
  Scaling correction factor                        NA

Parameter estimates:
Standard Errors                           Robust.sem
                  Estimate  Std.err  Z-value  P(>|z|)  Std.all
Regressions:
  masscat ~
    flood           -3.855    0.839   -4.595    0.000   -0.444


Thresholds:
    masscat|t1      -1.404    0.330   -4.262    0.000

R-square:
    masscat            0.197
```

34

Regression weight of -3.885 specifies the effect of one unit change in flood-level on the probability of observing mass_class = 1.

Error variance = 1.0 because it is set to that value to identify the model.

Note that separate modules on modeling with categorical outcomes will be developed to discuss more of the details related to their usage and interpretation.