The Stata Journal (*yyyy*) *vv*, Number *ii*, pp. 1–14

Combining information from multiple variables using models for causal indicators

Maarten L. Buis
Department of Sociology
Tübingen University
Tübingen, Germany
maarten.buis@uni-tuebingen.de

Abstract. One way in which one can better measure a variable, is to measure it more than once. However, after the different measures have been collected, one needs models that can combine the information from these different measurements. In this article I will introduce three such models: Sheaf coefficients, models with parametrically weighted covariates, and MIMIC models. What these models have in common is that they are models for so called 'causal indicators', that is, the observed variables are assumed to influence the underlying latent variable. Typical situations where these models can be useful occur when the observed variables can be thought of as resources adding up to a more general resource. For example, occupation and education of respondents adding up to socioeconomic status, or the amount of exercise and proportion of fruit in a diet adding up to the healthyness of the lifestyle. These models can also be used to scale the categories of a categorical explanatory variable such that the effect of that variable can be summarized by one number.

Keywords: st0001, latent variable, MIMIC, sheaf coefficients, parametrically weighted covariates, proportionality constraint

Sometimes we are in the fortunate position to have more than one measurement of the same concept. This is a good thing, as any measurement will contain error and measuring the same thing multiple times is a way of reducing this error. However, it also requires a model that can incorporate the additional information from the multiple measurements. There are roughly speaking two families of such models: First, there is set of models that can be derived from the assumption that the observed variables influence the latent variable. For example, we might observe a respondent's education and occupation and think that these influence that person's socioeconomic status (the latent variable). Second, there is a set of models that can be derived from the assumption that the latent variable influences the observed variables. For example, an intelligence test consisting of multiple questions is often based on the idea that the respondents latent intelligence influences that person's probability of answering the questions correctly. To distinguish between these two types of models, some authors, following Bollen (1984) and Bollen and Lennox (1991), call the observed variables "effect indicators" when they are influenced by the latent variable, and they call the observed variables "causal indicators" when they influence the latent variable. Distinguishing between these two is important as each requires a very different strategy for recovering the latent variable and its effect. Models for effect indicators are basically variations on the following idea:

one assumes that the only thing the indicators have in common is the latent variable, so any correlation between these variables must be due to the latent variable, and it is this correlation that is used to recover the latent variable. Models for causal indicators are based on the assumption that the latent variable is a weighted sum of the indicators (and optionally an error term), and the weights are estimated such that they are optimal for predicting the dependent variable. Models for effect indicators have already been implemented in Stata in the `factor` command ([R] **factor**) or in the models discussed by Hardouin (2007) and Kolenikov (2009), but models for causal indicators have been largely missing in Stata. An exception is that some of these models can be estimated using `gllamm` (Rabe-Hesketh et al. 2004; Skrondal and Rabe-Hesketh 2004)

In this article I will discuss three related models for causal indicators: a model with 'sheaf coefficients' (Heise 1972), a model with 'parametrically weighted covariates' (Yamaguchi 2002), and a Multiple Indicators and Multiple Causes (MIMIC) model (Hauser and Goldberger 1971). I will also discuss two programs — `sheafcoef` and `propcnsreg` — that can be used to estimate these models. First the idea behind the models will be discussed, followed by examples of how to estimate these in Stata, and a discussion of the syntax of the `sheafcoef` and `propcnsreg` packages. This article will end with a summary.

1 Sheaf coefficient

The sheaf coefficient is the simplest model of the three. Assume we want to explain a variable y using three observed variables x_1, x_2, and x_3, and we think that x_1 and x_2 actually influence y through a latent variable η and x_3 is a control variable. Because η is a latent variable, we need to fix its origin and its unit. The origin can be fixed by setting η to 0 when both x_1 and x_2 are 0, and the unit can be fixed by setting the standard deviation of η equal to 1. The model starts with a multiple regression model, where the βs are the regression coefficients and ε is a normally distributed error term, with a mean of 0 and a standard deviation that is to be estimated:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon \tag{1}$$

We want to turn this into equations 2 and 3, where λ is the effect of the latent variable and the γs are the effects of the observed variables on the latent variable:

$$\begin{aligned} y &= \beta_0 + \lambda\eta + \beta_3 x_3 + \varepsilon && (2)\\ \eta &= \gamma_0 + \gamma_1 x_1 + \gamma_2 x_2 && (3) \end{aligned}$$

We can fix the origin of η by constraining γ_0 to be 0. This way η will be 0 when both x_1 and x_2 equal 0. This leaves γ_1 and γ_2. We want to choose values for these parameters such that η optimally predicts y, and the standard deviation of η equals 1. This means that γ_1 and γ_2 are going to be a transformation of β_1 and β_2. We can start

with an initial guess that γ_1 equals β_1 and γ_2 equals β_2, and call the resulting latent variable η'. This will get us closer to where we want to be, as we now have values for all parameters: γ_0=0, γ_1'=β_1, γ_2'=β_2, and λ'=1. The value for λ' is derived from the fact that that is the only value where equations 2 and 3 lead to equation 1. However, the standard deviation of η' will generally not be equal to 1. The standard deviation of η' can be calculated as follows:

$$\text{sd}(\eta') = \sqrt{\beta_1^2\text{var}(x_1) + \beta_2^2\text{var}(x_2) + 2\beta_1\beta_2\text{cov}(x_1, x_2)} \tag{4}$$

We can recover η by dividing η' by its standard deviation, which means that the true values of γ_1 and γ_2 are actually $\beta_1/\text{sd}(\eta')$ and $\beta_2/\text{sd}(\eta')$. If we divide η' by its standard deviation, then we must multiply λ' by that same number to ensure that equations 2 and 3 continue to lead to equation 1. As a consequence λ will equal $\text{sd}(\eta')$.

Notice that this implies that the effect of the latent variable η is always positive. This is not restrictive as it seems, as what is high or low in η has not been fixed. So, to give a substantive interpretation of the direction of the effect of η one needs to look at the λs. For example, if x_1 is the proportion of vegetables in a person's diet, x_2 is the number of minutes spent a day exercising, and the λs are positive, then η would represent the 'healthyness' of a person's lifestyle. However, if the λs are negative, η would represent the 'unhealthyness' of a person's lifestyle. Alternatively, we could relax the constraint that λ must be positive by picking one of the observed variables and identify the direction of the latent variable relative to this observed variable by constrain η to have a high value when that observed variable has a high value and low when the observed variable has a low value (or exactly the opposite).

So, the key idea behind the sheaf coefficient (and all the models with causal indicators) is that if we think that the observed variables influence the latent variables, then that latent variable can be written as a weighted sum of the observed variables. The empirical information we use to recover the latent variable is that we choose those weights to maximize the effect of the latent variable on the dependent variable. Because we are trying to recover a latent variable we need the following three constraints:

- a constraint that fixes the origin of the latent variable.
- a constraint that fixes the unit of the latent variable.
- a constraint that either fixes the direction of latent variable or the direction of its effect.

One possible application of the sheaf coefficient is the comparison of effect sizes of different blocks of variables. For example, we may have a block of variables representing the family situation of the respondent and another block of variables representing characteristics of the work situation and we ask ourselves whether the work situation or the family situation is more important for determining a certain outcome variable. In that case we would estimate a model with two latent variables, one for the family situation

and one for the work situation, and since both latent variables are standardized their effects will be comparable.

This can be useful, but a sheaf coefficient merely reorders the information obtained from a regular regression. As a consequence, it is simply a different way of looking at the regression results, and it does not impose a testable constraint. Moreover, this model does not allow for any errors in the measurement of η, as equation 3 does not contain an error term.

1.1 Implementation in Stata: the `sheafcoef` pacakage

Sheaf coefficients can be estimated using the `sheafcoef` package. This package uses the fact that a sheaf coefficient is simply a transformation of coefficients, which allows it to be implemented as a post-estimation programme. This means that one must first estimate a model, using an estimation command like `regress` or `logit`, and then one can use `sheafcoef` to redisplay the results as a model with sheaf coefficients. It is therefore possible to use `sheafcoef` for many different kinds of dependent variables.

▷ Example

The use of this command can be illustrated using the `nlsw88.dta` dataset that comes with Stata. The first step is to open that dataset using the `sysuse` command, and prepare the variables.

```
. sysuse nlsw88, clear
(NLSW, 1988 extract)

.
. gen highschool = grade == 12 if grade < .
(2 missing values generated)
. gen somecollege = grade > 12 & grade < 16 if grade < .
(2 missing values generated)
. gen college = grade >= 16 if grade < .
(2 missing values generated)

.
. gen lnwage = ln(wage)

.
. gen ttl_exp2 = ttl_exp^2

.
. gen white = race == 1 if race < .
. gen other = race == 3 if race < .
```

In this example we have a set of dummies representing an individual's education (`highschool`, `somecollege`, and `college`, meaning that the reference category is those who have not finished high school), and a set of dummies representing an individual's race (`white`, and `other`, with African American as reference category), and we wonder which set of variables is more important for predicting an individuals wage while controlling for total experience in the labor market (`ttl_exp` and `ttl_exp2`). So, first

a regression of all these variables on log wage is estimated. After that, `sheafcoef` is used, specifying in the `latent()` option the blocks of variables that belong to the same latent variable preceded by the name we want to give to each latent variable. The name and the list of variables are separated using a colon (`:`), while the blocks of variables are separated using a semi-colon (`;`). So in this example, the education dummies are put in the first latent variable called `education` and the race dummies in the second latent variable called `race`. Notice that `college` is preceded by a `+`, this means that we identify the direction of the latent `education` variable relative to college such that `education` has a high value when a person has a college degree. None of the `race` dummies has a `+` or a `-` attached to it, so the direction of that latent variable is identified such that it has a positive effect on `lnwage`.

```
. reg lnwage white other ttl_exp ttl_exp2 highschool somecollege college
```

(*output omitted*)

```
. sheafcoef,                                                ///
>     latent(education: highschool somecollege +college ; ///
>            race:      white other)
```

lnwage	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
main						
education	.1898757	.0107139	17.72	0.000	.1688655	.2108859
race	.0517396	.0105663	4.90	0.000	.0310188	.0724603
ttl_exp	.0616495	.009803	6.29	0.000	.0424255	.0808734
ttl_exp2	-.0008656	.000395	-2.19	0.029	-.0016403	-.0000909
_cons	.9272152	.061262	15.14	0.000	.807079	1.047351
on_education						
highschool	.5726894	.1645151	3.48	0.001	.2500711	.8953078
somecollege	1.879124	.1570053	11.97	0.000	1.571233	2.187016
college	2.721446	.1063338	25.59	0.000	2.512923	2.929969
on_race						
white	2.283707	.0197364	115.71	0.000	2.245003	2.32241
other	2.086208	1.859547	1.12	0.262	-1.560411	5.732827

The parameters `education` and `race` in the `main` equation represent the effects of these latent variables on wage. The results show that education is more important than race for determining a person's income. The parameters in the `on_education` and `on_race` equations represent the effects of the dummies on the education latent variable and race latent variable respectively. These can be interpreted as scaling the different educational and race categories.

◁

2 Parametricaly weighted covariates

The model with parametricaly weighted covariates (Yamaguchi 2002) builds on the model with sheaf coefficients, but allows the effect of the latent variable to change over one or more other variables. This means that equations 5 and 6, where the effect of η

changes over x_3 will be estimated, instead of equations 2 and 3.

$$\begin{aligned} y &= \beta_0 + (\lambda_0 + \lambda_1 x_3)\eta + \beta_3 x_3 + \varepsilon \quad (5) \\ \eta &= \gamma_0 + \gamma_1 x_1 + \gamma_2 x_2 \quad (6) \end{aligned}$$

If η is replaced by equation 6, and the origin of η is fixed by constraining γ_0 to be zero, we get:

$$\begin{aligned} y &= \beta_0 + (\lambda_0 + \lambda_1 x_3)(\gamma_1 x_1 + \gamma_2 x_2) + \beta_3 x_3 + \varepsilon \\ &= \beta_0 + (\lambda_0 + \lambda_1 x_3)\gamma_1 x_1 + (\lambda_0 + \lambda_1 x_3)\gamma_2 x_2 + \beta_3 x_3 + \varepsilon \end{aligned}$$

This means the effect of x_1 (through η) on y equals $(\lambda_0+\lambda_1 x_3)\gamma_1$, and that the effect of x_2 (through η) on y equals $(\lambda_0 + \lambda_1 x_3)\gamma_2$. This implies the following constraint: for every value of x_3, the effect of x_1 relative to x_2 will always be $(\lambda_0 + \lambda_1 x_3)\gamma_1/(\lambda_0 + \lambda_1 x_3)\gamma_2 = \gamma_1/\gamma_2$, which is a constant. In other words, the model with parametricaly weighted covariates imposes a proportionality constraint. The empirical information used to estimate this model is still that we choose the weights (γs) to maximize the effect of the latent variable on the dependent variable, but now we add this proportionality constraint.

There are now a variety of ways in which we could identify the unit of the latent variable: We could use the same strategy as with sheaf coefficients, that is, identify the unit of the latent variable by setting its standard deviation to 1 and identify the direction either such that the effect of the latent variable is positive or relative to one of the observed variables. Alternatively, the coefficient λ_0 can be set to 1, which means that γ_1 and γ_2 represent the indirect effects of x_1 and x_2 through the latent variable on y when x_3 equals 0, or either the coefficient γ_1 or γ_2 can be set to 1, which means that the unit of the latent variable will equal the unit of x_1 or x_2 respectively.

The proportionality constraint can also be of substantive interest without referring to a latent variable. Consider a model where one wants to explain the respondent's education with the eduction of the father (fed) and the mother (med), and that one is interested in testing whether the relative contribution of the mother's education has increased over time. The model with parametrically weighted covariates will estimate this model under the null hypothesis that effects of fed and med might have changed over time, but that the size of the effect of med relative to the size of fed has remained constant.

2.1 Implementation in Stata: the `propcnsreg` package

The `propcnsreg` package can estimate a model with parametricaly weighted covariates. Unlike the models with sheaf coefficients, these models need to be separately estimated,

and can thus not be implemented as a post-estimation command like `sheafcoef`. This means that `propcnsreg` can only be used for continuous dependent variables with (approximately) normally distributed errors.

▷ Example

The use of `propcnsreg` can be illustrated by continuing the example. Now we assume that the effect of education changes for different levels of experience. The parameters in the `constrained` panel represent the scale of the latent education variable. By specifying the `unit(college)` option, we identified the unit of the latent education variable such that parameters of `highschool` and `somecollege` represent the positions of these levels relative to less than high school (0) and college (1). The parameters in the panel `lambda` represent the effect of the latent optimally-scaled education on `lnwage` and how it changes when experience changes. The `unconstrained` panel shows the main effects of experience and the control variables. A test of the proportionality constraint is reported at the bottom of the output.

```
. propcnsreg lnwage white other ttl_exp ttl_exp2, /*
>         */ lambda(ttl_exp ttl_exp2) /*
>         */ constrained(highschool somecollege college) /*
>         */ unit(college) nolog
                                                Number of obs   =       2244
                                                LR chi2(10)     =     101.57
Log likelihood = -1573.1308                     Prob > chi2     =     0.0000

Constraint: [constrained]college = 1
```

lnwage	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
unconstrai~d						
white	.1166583	.0240475	4.85	0.000	.0695259	.1637906
other	.1101377	.0981958	1.12	0.262	-.0823226	.3025981
ttl_exp	.0372122	.0041246	9.02	0.000	.0291281	.0452963
ttl_exp2	.0006399	.0006602	0.97	0.332	-.0006541	.0019339
_cons	1.516685	.0362236	41.87	0.000	1.445688	1.587682
constrained						
highschool	.2431709	.0550686	4.42	0.000	.1352385	.3511033
somecollege	.7056825	.0538163	13.11	0.000	.6002045	.8111604
college	1	.	.	.	.	.
lambda						
ttl_exp	.0097898	.0072732	1.35	0.178	-.0044653	.024045
ttl_exp2	-.0039162	.0012162	-3.22	0.001	-.0062999	-.0015325
_cons	.5989646	.0440969	13.58	0.000	.5125363	.6853928
ln_sigma						
_cons	-.7178998	.014927	-48.09	0.000	-.7471563	-.6886434

```
LR test vs. unconstrained model: chi2(4) =     13.31   Prob > chi2 =     0.010
```

◁

3 MIMIC

The MIMIC model builds on the model with parametricaly weighted covariates by assuming that the latent variable is measured with error. This means that the model in equations 7 and 8 is estimated. The only difference with the model with parametrically weighted covariates is the addition of the error term ε_η in equation 8.

$$y = \beta_0 + (\lambda_0 + \lambda_1 x_3)\eta + \beta_3 x_3 + \varepsilon_y \quad (7)$$

$$\eta = \gamma_1 x_1 + \gamma_2 x_2 + \varepsilon_\eta \quad (8)$$

The ε_y and ε_η are independent normally distributed error terms with means zero and standard deviations that need to be estimated. By replacing η in equation 7 by equation 8, one can see that the error term of this model is:

$$\varepsilon_y + (\lambda_0 + \lambda_1 x_3)\varepsilon_\eta \quad (9)$$

This combined error term will also be normally distributed, as the sum of two independent normally distributed variables is itself also normally distributed. The mean of this combined error term will be zero and it will have the following standard deviation:

$$\sqrt{\text{var}(\varepsilon_y) + (\lambda_0 + \lambda_1 x_3)^2 \text{var}(\varepsilon_\eta)} \quad (10)$$

So the empirical information that is used to separate the standard deviation of ε_y from the standard deviation of ε_η, is the changes in the residual variance over x_3. The data will thus contain rather indirect information that can be used for estimating this part of the model. However, if the model is correct, it will make it possible to control for measurement error in the latent variable.

There is an important downside to this model, and that is that heteroscedasticity, and in particular changes in the variance of ε_y over x_3, could have a distorting influence on the parameter estimates of λ_0 and λ_1. Consider again the example of wanting to explain the respondent's education through the education of the father and the mother, but now assume that we are interested in how the effect of the latent parental education variable changes over time. In this case we have good reason to suspect that the variance of ε_y will also change over time: education consists of a discrete number of categories, and in early cohorts most of the respondents tend to cluster in the lowest categories. Over time the average level of education tends to increase, which in practice means that the respondents tend to cluster less in the lowest category, and have more room to differ from one another. As a consequence the residual variance is likely to have increased over cohorts. Normally this heteroscedasticity would not be an issue of great concern, but in a MIMIC model this heteroscedasticity is incorrectly interpreted as indicating that there is measurement error in the latent variable representing parental education. Moreover, this 'information' on the measurement error is used to 'refine' the estimates

of λ_0 and λ_1. So, this would be an example where the MIMIC model would not be appropriate.

3.1 Implementation in Stata: the `propcnsreg` package with the `mimic` option

A MIMIC model can be estimated using `propcnsreg` by specifying the `mimic` option. This means that an extra parameter (`ln_sigma_latent`) is estimated representing the log of the standard deviation of the measurement error of the latent variable.

▷ Example

Continuing the example, we can estimate the following MIMIC model. Notice that the test for the proportionality constraint is no longer reported, as it is unclear how to identify the error variance of the latent variable without the proportionality constraint.

```
. propcnsreg lnwage white other ttl_exp ttl_exp2, /*
>         */ lambda(ttl_exp ttl_exp2) /*
>         */ constrained(highschool somecollege college) /*
>         */ unit(college) mimic nolog
                                                Number of obs   =       2244
                                                LR chi2(10)     =     135.25
Log likelihood = -1571.1459                     Prob > chi2     =     0.0000

Constraint: [constrained]college = 1
```

lnwage	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
unconstrai~d						
white	.1187835	.0240302	4.94	0.000	.071685	.1658819
other	.101796	.0970624	1.05	0.294	-.0884427	.2920347
ttl_exp	.039887	.0036913	10.81	0.000	.0326521	.0471218
ttl_exp2	.0009142	.0006198	1.48	0.140	-.0003006	.0021291
_cons	1.517155	.0366785	41.36	0.000	1.445266	1.589043
constrained						
highschool	.2304432	.0556791	4.14	0.000	.1213142	.3395722
somecollege	.7022772	.0550723	12.75	0.000	.5943375	.8102169
college	1	.	.	.	.	.
lambda						
ttl_exp	.0039014	.0067611	0.58	0.564	-.0093502	.0171529
ttl_exp2	-.00456	.0011831	-3.85	0.000	-.0068787	-.0022412
_cons	.6044364	.0446813	13.53	0.000	.5168626	.6920101
ln_sigma						
_cons	-.8121774	.050151	-16.19	0.000	-.9104716	-.7138833
ln_sigma_l~t						
_cons	-.9502239	.2506964	-3.79	0.000	-1.44158	-.458868

◁

4 Syntax and options

Syntax of `sheafcoef`

`sheafcoef, latent(latent_spec ) [ eform equation(##| name) post level(#) ]`

Options of `sheafcoef`

`latent(`*latent_spec*`)` specifies the blocks of variables that make up the latent variables. The syntax for *latent_spec* is:

`[`*name_1*`:][+|-]`*var_1* `[+|-]`*var_2* `...` `[; [`*name_2*`:][+|-]`*var_3* `[+|-]`*var_4* `...] [...]`

Each block is separated by a semicolon (`;`). Each block needs to consist of at least two variables. These variables must be explanatory variables in the estimation command preceding `sheafcoef`, and the same variable can only appear in one block. Each block can be preceded by a name for the latent variable it represents. The name and the variables are separated by a colon (`:`). Moreover, one can identify one key variable in each block of variables by attaching a `+` or `-` (without a space) to a variable in a block. If one of the observed variables has `+` attached to it, then the latent variable will have a high value when that observed variable is high and a low variable when that observed variable is low. The opposite is true when one of the observed variables has a `-` attached to it. If no observed variable in a block has a `+` or a `-` attached to it, then the direction of that latent variable is identified such that it's effect on the dependent variable is positive.

`equation(`*## | name*`)` specifies the equation from the previous estimation command to be used when computing the sheaf coefficients. This option is relevant when using `sheafcoef` after commands like [R] **mlogit** or [R] **heckman** that return results in multiple equations. One can either specify whether `sheafcoef` should consider the first, second, etc. equation or one can type in the name of that equation. In the former case the number of the equation should be preceded by a #.

`eform` specifies that the effects of the latent variable and the control variables are exponentiated. The effects of the indicator variables in each block on its latent variable are not exponentiated because these represent the effects of these variables on the standardized latent variable and not on the dependent variable. This option can be useful after commands like `logit` or `poisson`, as this will cause the effects on the dependent variables to be displayed in the form of odds ratios and incidence rate ratios respectively.

`post` causes `sheafcoef` to behave like a Stata estimation (e-class) command. When `post` is specified, `sheafcoef` will post the vector of transformed estimators and its estimated variance-covariance matrix to `e()`. This option, in essence, makes the transformation permanent. Thus you could, after posting, treat the transformed

estimation results in the same way as you would treat results from other Stata estimation commands. For example, after posting, you could use [R] **test** to perform simultaneous tests of hypotheses on linear combinations of the transformed estimators.

Specifying `post` clears the previous estimation results, which can then only be recovered by refitting the original model or by storing the estimation results before running `sheafcoef` and then restoring them; see [R] **estimates store**.

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`, see [R] **level**.

Syntax of `propcnsreg`

`propcnsreg` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] `,` `constrained(`*varlist*`)` `lambda(`*varlist*`)` [`standardized` `lcons` `unit(`*varname*`)` `mimic` `robust` `cluster(`*varname*`)` `level(#)` *em_maximize_options* *maximize_options*]

Options of `propcnsreg`

`constrained(`*varlist_c*`)` specifies the variables that are measurements of the latent variable. The effects of these variables are to be constrained to change by the same proportion as the variables specified in `lambda()` change. One can precede one of these variables with either a `+` or a `-` to a variable (without a space) to indicate that the direction of the latent variable is to be identified relative to this observed variable if the `standardized` option is specified. If the observed variable is preceded by a `+` then the latent variable has a high value when that observed variable has a high value, and when the observed variable is preceded by a `-` then the latent variable has a low value when that observed variable has a high value.

`lambda(`*varlist_l*`)` specifies the variables along which the effects of the latent variable changes.

`standardized` specifies that the unit of the latent variable is identified by constraining the standard deviation of the latent variable to be equal to 1. This is the default parametrization.

`lcons` specifies that the unit of the latent variable is identified by setting the constant of the `lambda` equation equal to 1. This means that the parameters of the variables specified in the option `constrained()` measure the indirect effect of these variables through the latent variable on the dependent variable when all variables specified in the option `lambda()` are zero.

`unit(`*varname*`)` specifies that the scale of the latent variable is identified by constraining the unit of the latent variable to be equal to the unit of *varname*. The variable *varname* must be specified in `constrained()` option.

`mimic` specifies that a MIMIC model is to be estimated.

`robust` specifies that the Huber/White/sandwich estimator of variance is to be used instead of the traditional calculation; see [U] **23.14 Obtaining robust variance estimates**. `robust` combined with `cluster()` allows observations which are not independent within cluster (although they must be independent between clusters).

`cluster(`*clustervar*`)` specifies that the observations are independent across groups (clusters) but not necessarily within groups. *clustervar* specifies to which group each observation belongs; e.g., `cluster(`*personid*`)` in data with repeated observations on individuals. See [U] **23.14 Obtaining robust variance estimates**. Specifying `cluster()` implies `robust`.

`level(#)` specifies the confidence level, in percent, for the confidence intervals of the coefficients; see [R] **level**.

em_maximize_options

The starting values are determined by a number of iterations of an EM algorithm, and the following options allows one to fine tune this algorithm. These options are seldom used.

`emiterate(#)` specifies the maximum number of iterations for the EM algorithm. When the number of iterations equals `emiterate()`, the EM algorithm stops. If convergence is declared before this threshold is reached, it will stop when convergence is declared. The default value of `emiterate()` is 20.

`emtolerance(#)` specifies the tolerance for the coefficient vector. When the relative change in the coefficient vector from one iteration to the next is less than or equal to `emtolerance()`, the `emtolerance()` convergence criterion is satisfied. `emtolerance(1e-6)` is the default.

`emltolerance(#)` specifies the tolerance for the log likelihood. When the relative change in the log likelihood from one iteration to the next is less than or equal to `emltolerance()`, the `emltolerance()` convergence is satisfied. `emltolerance(1e-7)` is the default

maximize_options

`difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `trace`, `gradient`, `showstep`, `hessian`, `shownrtolerance`, `tolerance(#)`, `ltolerance(#)`, `gtolerance(#)`, `nrtolerance(#)`, `nonrtolerance(#)`; see [R] **maximize**. These options are seldom used.

5 Summary

In this article I discussed three models for combining information from multiple measurements of the same variable: the model with sheaf coefficients, the model with para-

metrically weighted covariates, and the MIMIC model. These are models for so called 'causal indicators', which means that the observed variables are thought to influence the underlying latent variable. They are an alternative for models for so called 'effect indicators' where the observed variables are though to be influenced by the underlying latent variable, like factor analysis.

The models for causal indicators use the following strategy to recover the underlying latent variable: the latent variable is thought of as a weighted sum of the observed variables, where the weights are the effects of the observed variables on the latent variable. The weights are chosen to maximize the effect of the latent variable on the dependent variable, subject to at least the following identifying constraints:

- a constraint that fixes the origin,
- a constraint that fixes the unit, and
- a constraint that fixes the direction of either the latent variable or it's effect.

The model using sheaf coefficients only uses this minimal set of identifying constraints. This means that a sheaf coefficient does not imply a testable constraint, it is just a different way of representing the results from a regular model. The model with parametrically weighted covariates allows the effect of the latent variable to change over another variable. It uses the same strategy as models with sheaf coefficients to recover the latent variable, but adds the testable constraint that the relative sizes of the effects of the observed indicators through the latent variable on the dependent variable does not change over this other variable. The MIMIC model builds on the model with parametrically weighted covariates but allows the latent variable to be measured with error. It uses the same strategy as the model with parametrically weighted covariates to recover the latent variable, but it adds the constraint that total residual variance depends on the variables along which the latent variable is allowed to change.

I also discussed the way in which these models are implemented in Stata. The model with sheaf coefficients is implemented as the `sheafcoef` package. The model with parametrically weighted covariates and the MIMIC model are implemented in the `propcnsreg` package.

6 Acknowledgement

I thank Richard T. Campbell for literature references and encouragements.

7 References

Bollen, K. A. 1984. Multiple indicators: Internal consistency or no necessary relationship? *Quality & Quantity* 18(4): 399–385.

Bollen, K. A., and R. Lennox. 1991. Conventional wisdom on measurement: A structural equation perspective. *Psychological Bulletin* 110(2): 305–314.

Hardouin, J.-B. 2007. Rasch analysis: Estimation and tests with raschtest. *The Stata Journal* 7: 22–44.

Hauser, R. M., and A. S. Goldberger. 1971. The treatment of unobservable variables in path analysis. *Sociological Methodology* 3: 81–117.

Heise, D. R. 1972. Employing nominal variables, induced variables, and block variables in path analysis. *Sociological Methods & Research* 1(2): 147–173.

Kolenikov, S. 2009. Confirmatory factor analysis using confa. *The Stata Journal* 9: 329–373.

Rabe-Hesketh, S., A. Skrondal, and A. Pickles. 2004. Generalized multilevel structural equation modelling. *Psychometrika* 69: 167–190.

Skrondal, A., and S. Rabe-Hesketh. 2004. *Generalized Latent Variable Modeling: Multilevel, Longitudinal, and Structural Equation Models*. Boca Raton: Chapman & Hall/CRC.

Yamaguchi, K. 2002. Regression models with parametrically weighted explanatory variables. *Sociological Methodology* 32: 219–245.

About the author

Maarten L. Buis is affiliated with the Department of Sociology of the University Tübingen, where he studies the interaction between demographic processes and inequality of educational opportunity. He is also a frequent contributor to statalist.