Syntax

mmme(data, prox.loc, w=”Identity Matrix”, latent.names=NULL, opt=TRUE, report.contr=FALSE, var.first=TRUE,cluster=FALSE,disp=TRUE)

Arguments

data	A data matrix in which the dependent variable is in the first column, followed by blocks of columns of the proxies for each latent variable and the controls in the last columns.
prox.loc	A vector containing first the locations of the first proxy for each latent variable and as the last entry the column of the first control (i.e. c(2,6,9) for two latent variables with 4 and 3 proxies)
prox.no	A vector containing the number of proxies for the latent variables (i.e. c(4,3) for two latent variables with 4 and 3 proxies)

Options

w	Allows specification of a first step weighting matrix. Optimal weighting will be used in second step. The default is the identity matrix.
latent.names	Vector of length of number of latent variables that gives the names of the latent dimensions. Default is NULL, in which case the latent dimensions will be given consecutive numbers.
opt	If FALSE, estimation will stop after the first step and standard GMM rather than OGMM estimates are returned. Default is TRUE
report.contr	If TRUE, all coefficients from yulizing step are reportet. If FALSE (default), only coefficients in outcome equation are reported.
var.first	If FALSE, first step variance will not be computed for optimal GMM, which is only used for the Hausman-Test. Default is TRUE.
cluster	Column number of the clustering variable. Default is FALES, i.e. observations are assumed to be iid.
disp	If FALSE, no output is displayed, but the results are returned. Default is TRUE.

Items Saved

The program returns a list with the following entries:

$Stats:

$stats$N: Number of Observations
$stats$np: Number of latent variables
$stats$nm: Vector with the number of proxies for each latent variable
$stats$oir: p-value of overidentification restrictions test
$stats$waldp: p-value of Wald test that all coefficients are 0
$stats$hausp: p-value of Hausman test that optimally weighted GMM is unbiased

$first: results from first step minimization. Output of nlm(), contains:

$first$minimum: value of loss function at minimum
$first$estimate: parameter values
$gradient: gradient at minimum
$code: reason why iterations terminated. See ?nlm for interpretation
$iterations: Number of iterations of minimization routine

$opt: results from optimally weighted minimization. See $first for details.

$varfirst: Variance-Covariance matrix of the first step estimator (only if var.first=TRUE)

$varopt: Variance-Covariance matrix of the optimally weighted GMM estimator

Note on Hausman Test

Besides the usual caution with which Hausman Tests should be used, there is an additional problem here, since the direction of the scales of the latent variables is not identified and is left for the minimizing algorithm to choose. One may therefore end up with all parameters except for the variances associated with one latent variable changing signs from the GMM to the OGMM step. For the Hausman test, the signs of the parameters of the GMM step are forced to have the same sign as the OGMM coefficients. If some coefficients are relatively close to zero, however, their signs may be different in the two steps because of sampling variability and not because the scale is inverted. In this case, they should not be coerced to have the same sign, and the program will produce a wrong test statistic. If there are coefficients close to zero, one should thus not rely on the Hausman Test without checking whether the signs of the estimates of the two steps are congruent except for inversion of the scales.

Postestimation

Wald Tests

A Wald Test whether all coefficients are jointly zero is included in the output. Wald Tests of subsets of the coefficients can be performed using the saved Variance-Covariance Matrix and parameters if a package that includes a Wald test is loaded. E.g. with package “aod” loaded, a Wald test of H0: coefficients (1,4,5) are jointly zero can be performed using:

```
wald.test(<results>$varopt,<results>$opt$estimates,Terms=c(1,4,5))
```

See ?wald.test for more details.

Recover implied OLS slopes

recoverols(param,prox.loc,data, bootstrap=FALSE, bsiter=1000,cluster=FALSE)

Calculates implied and actual OLS coefficient, the SE of the OLS coefficient and the absolute value of the difference. The function returns a list in which the results are saved as $results.

If bootstrap is TRUE, the SEs of the difference between the coefficients are bootstrapped using the number of iterations specified in bsiter. The distribution of the difference is saved in $dist.bs, which should be checked for NAs, since these occur when the minimization may have failed to converge. $failure additionally contains the number of times convergence failed. Note that I consider it a failure of convergence if nlm returns code 3, which in most of my test runs did identify the minimum, but may potentially not.

Predict Factor Scores

Predscores(param,prox.loc,data,scale=0)

Calculates factor scores based on the method proposed by Bartlett. Since the scale of the scores is not identified, scale can be used to change it. Its main use probably is to change the sign of the scale, in which case it should be -1 if the scale of all proxies should be reversed or a vector with as many entries as there are proxies, each giving the factor by which the scale should be multiplied.

Problems:

I frequently run into some problems with the convergence of the first step estimator, particularly when dealing with more than two latent variables. It is usually quite obvious from the results that the algorithm did not converge. I have experimented extensively with 2 and 3 latent variables and 4-5 proxies each. Other configurations should work, but it may be worth double checking the results. I think this was the first program I ever wrote (certainly the first in R), so I'm sure it is not written very elegantly, but it seems to do the work. If you have any problems or notice any mistakes, please let me know (mittag@uchicago.edu).