Title

stata.com

mgarch dvech — Diagonal vech multivariate GARCH models

Description

`mgarch dvech` estimates the parameters of diagonal vech (DVECH) multivariate generalized autoregressive conditionally heteroskedastic (MGARCH) models in which each element of the conditional correlation matrix is parameterized as a linear function of its own past and past shocks.

DVECH MGARCH models are less parsimonious than the conditional correlation models discussed in [TS] **mgarch ccc**, [TS] **mgarch dcc**, and [TS] **mgarch vcc** because the number of parameters in DVECH MGARCH models increases more rapidly with the number of series modeled.

Quick start

Fit diagonal vech multivariate GARCH with first- and second-order ARCH components for dependent variables `y1` and `y2` using `tsset` data

```
mgarch dvech (y1 y2), arch(1 2)
```

Add regressors `x1` and `x2` and first-order GARCH component

```
mgarch dvech (y1 y2 = x1 x2), arch(1 2) garch(1)
```

Menu

Statistics > Multivariate time series > Multivariate GARCH

Syntax

```
mgarch dvech eq [eq ... eq] [if] [in] [, options]
```

where each *eq* has the form

```
(depvars = [indepvars] [, noconstant])
```

options	Description
Model	
`arch(`*numlist*`)`	ARCH terms
`garch(`*numlist*`)`	GARCH terms
`distribution(`*dist* `[#]``)`	use *dist* distribution for errors [may be `gaussian` (synonym `normal`) or `t`; default is `gaussian`]
`constraints(`*numlist*`)`	apply linear constraints
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim` or `robust`
Reporting	
`level(#)`	set confidence level; default is `level(95)`
`nocnsreport`	do not display constraints
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
`from(`*matname*`)`	initial values for the coefficients; seldom used
`svtechnique(`*algorithm_spec*`)`	starting-value maximization algorithm
`sviterate(#)`	number of starting-value iterations; default is `sviterate(25)`
`coeflegend`	display legend instead of statistics

You must `tsset` your data before using `mgarch dvech`; see [TS] **tsset**.

indepvars may contain factor variables; see [U] **11.4.3 Factor variables**.

depvars and *indepvars* may contain time-series operators; see [U] **11.4.4 Time-series varlists**.

`by`, `collect`, `fp`, `rolling`, and `statsby` are allowed; see [U] **11.1.10 Prefix commands**.

`coeflegend` does not appear in the dialog box.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

Model

`noconstant` suppresses the constant term(s).

`arch(`*numlist*`)` specifies the ARCH terms in the model. By default, no ARCH terms are specified.

`garch(`*numlist*`)` specifies the GARCH terms in the model. By default, no GARCH terms are specified.

`distribution(`*dist* `[#]``)` specifies the assumed distribution for the errors. *dist* may be `gaussian`, `normal`, or `t`.

`gaussian` and `normal` are synonyms; each causes `mgarch dvech` to assume that the errors come from a multivariate normal distribution. # cannot be specified with either of them.

`t` causes `mgarch dvech` to assume that the errors follow a multivariate Student t distribution, and the degree-of-freedom parameter is estimated along with the other parameters of the model. If `distribution(t #)` is specified, then `mgarch dvech` uses a multivariate Student t distribution with # degrees of freedom. # must be greater than 2.

`constraints(`*numlist*`)` specifies linear constraints to apply to the parameter estimates.

SE/Robust

`vce(`*vcetype*`)` specifies the estimator for the variance–covariance matrix of the estimator.

`vce(oim)`, the default, specifies to use the observed information matrix (OIM) estimator.

`vce(robust)` specifies to use the Huber/White/sandwich estimator.

Reporting

`level(#)`, `nocnsreport`; see [R] **Estimation options**.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, `nonrtolerance`, and `from(`*matname*`)`; see [R] **Maximize** for all options except `from()`, and see below for information on `from()`. These options are seldom used.

`from(`*matname*`)` specifies initial values for the coefficients. `from(b0)` causes `mgarch dvech` to begin the optimization algorithm with the values in `b0`. `b0` must be a row vector, and the number of columns must equal the number of parameters in the model.

`svtechnique(`*algorithm_spec*`)` and `sviterate(#)` specify options for the starting-value search process.

`svtechnique(`*algorithm_spec*`)` specifies the algorithm used to search for initial values. The syntax for *algorithm_spec* is the same as for the `technique()` option; see [R] **Maximize**. `svtechnique(bhhh 5 nr 16000)` is the default. This option may not be specified with `from()`.

`sviterate(#)` specifies the maximum number of iterations that the search algorithm may perform. The default is `sviterate(25)`. This option may not be specified with `from()`.

The following option is available with `mgarch dvech` but is not shown in the dialog box:

`coeflegend`; see [R] **Estimation options**.

Remarks and examples

stata.com

We assume that you have already read [TS] **mgarch**, which provides an introduction to MGARCH models and the methods implemented in `mgarch dvech`.

MGARCH models are dynamic multivariate regression models in which the conditional variances and covariances of the errors follow an autoregressive-moving-average structure. The DVECH MGARCH model parameterizes each element of the current conditional covariance matrix as a linear function of its own past and past shocks.

As discussed in [TS] **mgarch**, MGARCH models differ in the parsimony and flexibility of their specifications for a time-varying conditional covariance matrix of the disturbances, denoted by $\mathbf{H}_t$. In a DVECH MGARCH model with one ARCH term and one GARCH term, the (i, j)th element of conditional covariance matrix is modeled by

$$h_{ij,t} = s_{ij} + a_{ij}\boldsymbol{\epsilon}_{i,t-1}\boldsymbol{\epsilon}_{j,t-1} + b_{ij}h_{ij,t-1}$$

where s_{ij}, a_{ij}, and b_{ij} are parameters and $\boldsymbol{\epsilon}_{t-1}$ is the vector of errors from the previous period. This expression shows the linear form in which each element of the current conditional covariance matrix is a function of its own past and past shocks.

❑ Technical note

The general vech MGARCH model developed by Bollerslev, Engle, and Wooldridge (1988) can be written as

$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \boldsymbol{\epsilon}_t \tag{1}$$

$$\boldsymbol{\epsilon}_t = \mathbf{H}_t^{1/2}\boldsymbol{\nu}_t \tag{2}$$

$$\mathbf{h}_t = \mathbf{s} + \sum_{i=1}^{p}\mathbf{A}_i\text{vech}(\boldsymbol{\epsilon}_{t-i}\boldsymbol{\epsilon}_{t-i}') + \sum_{j=1}^{q}\mathbf{B}_j\mathbf{h}_{t-j} \tag{3}$$

where

$\mathbf{y}_t$ is an $m \times 1$ vector of dependent variables;

$\mathbf{C}$ is an $m \times k$ matrix of parameters;

$\mathbf{x}_t$ is a $k \times 1$ vector of independent variables, which may contain lags of $\mathbf{y}_t$;

$\mathbf{H}_t^{1/2}$ is the Cholesky factor of the time-varying conditional covariance matrix $\mathbf{H}_t$;

$\boldsymbol{\nu}_t$ is an $m \times 1$ vector of independent and identically distributed innovations;

$\mathbf{h}_t = \text{vech}(\mathbf{H}_t)$;

the vech() function stacks the lower diagonal elements of a symmetric matrix into a column vector, for example,

$$\text{vech}\begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} = (1,\ 2,\ 3)'$$

$\mathbf{s}$ is an $m(m+1)/2 \times 1$ vector of parameters;

each $\mathbf{A}_i$ is an $\{m(m+1)/2\} \times \{m(m+1)/2\}$ matrix of parameters; and

each $\mathbf{B}_j$ is an $\{m(m+1)/2\} \times \{m(m+1)/2\}$ matrix of parameters.

Bollerslev, Engle, and Wooldridge (1988) argued that the general-vech MGARCH model in (1)–(3) was too flexible to fit to data, so they proposed restricting the matrices $\mathbf{A}_i$ and $\mathbf{B}_j$ to be diagonal matrices. It is for this restriction that the model is known as a diagonal vech MGARCH model. The diagonal vech MGARCH model can also be expressed by replacing (3) with

$$\mathbf{H}_t = \mathbf{S} + \sum_{i=1}^{p} \mathbf{A}_i \odot \boldsymbol{\epsilon}_{t-i}\boldsymbol{\epsilon}'_{t-i} + \sum_{j=1}^{q} \mathbf{B}_j \odot \mathbf{H}_{t-j} \qquad (3')$$

where $\mathbf{S}$ is an $m \times m$ symmetric parameter matrix; each $\mathbf{A}_i$ is an $m \times m$ symmetric parameter matrix; $\odot$ is the elementwise or Hadamard product; and each $\mathbf{B}_j$ is an $m \times m$ symmetric parameter matrix. In (3′), $\mathbf{A}$ and $\mathbf{B}$ are symmetric but not diagonal matrices because we used the Hadamard product. The matrices are diagonal in the vech representation of (3) but not in the Hadamard-product representation of (3′).

The Hadamard-product representation in (3′) clarifies that each element in $\mathbf{H}_t$ depends on its past values and the past values of the corresponding ARCH terms. Although this representation does not allow cross-covariance effects, it is still quite flexible. The rapid rate at which the number of parameters grows with m, p, or q is one aspect of the model's flexibility.

❑

Some examples

▷ Example 1: Model with common covariates

We have data on a secondary market rate of a six-month U.S. Treasury bill, `tbill`, and on Moody's seasoned AAA corporate bond yield, `bond`. We model the first differences of `tbill` and the first differences of `bond` as a VAR(1) model with an ARCH(1) term.

```
. use https://www.stata-press.com/data/r18/irates4
(St. Louis Fed (FRED) financial data)

. mgarch dvech (D.bond D.tbill = LD.bond LD.tbill), arch(1)

Getting starting values
(setting technique to bhhh)
Iteration 0:  Log likelihood =  3569.2723
Iteration 1:  Log likelihood =  3708.4561
  (output omitted)
Iteration 6:  Log likelihood =  4183.8853
Iteration 7:  Log likelihood =  4184.2424
(switching technique to nr)
Iteration 8:  Log likelihood =  4184.4141
Iteration 9:  Log likelihood =  4184.5973
Iteration 10: Log likelihood =  4184.5975

Estimating parameters
(setting technique to bhhh)
Iteration 0:  Log likelihood =  4184.5975
Iteration 1:  Log likelihood =  4200.6303
Iteration 2:  Log likelihood =  4208.5342
Iteration 3:  Log likelihood =   4212.426
Iteration 4:  Log likelihood =  4215.2373
(switching technique to nr)
Iteration 5:  Log likelihood =  4217.0676
Iteration 6:  Log likelihood =  4221.5706
Iteration 7:  Log likelihood =  4221.6576
Iteration 8:  Log likelihood =  4221.6577
```

```
Diagonal vech MGARCH model
Sample: 3 thru 2456                                   Number of obs =   2,454
Distribution: Gaussian                                Wald chi2(4)  = 1183.52
Log likelihood = 4221.658                             Prob > chi2   =  0.0000
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
D.bond						
bond						
LD.	.2967674	.0247149	12.01	0.000	.2483271	.3452077
tbill						
LD.	.0947949	.0098683	9.61	0.000	.0754533	.1141364
_cons	.0003991	.00143	0.28	0.780	-.0024036	.0032019
D.tbill						
bond						
LD.	.0108373	.0301501	0.36	0.719	-.0482558	.0699304
tbill						
LD.	.4344747	.0176497	24.62	0.000	.3998819	.4690675
_cons	.0011611	.0021033	0.55	0.581	-.0029612	.0052835
/Sigma0						
1_1	.004894	.0002006	24.40	0.000	.0045008	.0052871
2_1	.0040986	.0002396	17.10	0.000	.0036289	.0045683
2_2	.0115149	.0005227	22.03	0.000	.0104904	.0125395
L.ARCH						
1_1	.4514942	.0456835	9.88	0.000	.3619562	.5410323
2_1	.2518879	.036736	6.86	0.000	.1798866	.3238893
2_2	.843368	.0608055	13.87	0.000	.7241914	.9625446

The output has three parts: an iteration log, a header, and an output table. The iteration log has two parts: the first part reports the iterations from the process of searching for starting values, and the second part reports the iterations from maximizing the log-likelihood function.

The header describes the estimation sample and reports a Wald test against the null hypothesis that all the coefficients on the independent variables in each equation are zero. Here the null hypothesis is rejected at all conventional levels.

The output table reports point estimates, standard errors, tests against zero, and confidence intervals for the estimated coefficients, the estimated elements of $\mathbf{S}$, and any estimated elements of $\mathbf{A}$ or $\mathbf{B}$. Here the output indicates that in the equation for `D.tbill`, neither the coefficient on `LD.bond` nor the constant are statistically significant. The elements of $\mathbf{S}$ are reported in the `Sigma0` equation. The estimate of $\mathbf{S}[1,1]$ is 0.005, and the estimate of $\mathbf{S}[2,1]$ is 0.004. The ARCH term results are reported in the `L.ARCH` equation. In the `L.ARCH` equation, `1_1` is the coefficient on the ARCH term for the conditional variance of the first dependent variable, `2_1` is the coefficient on the ARCH term for the conditional covariance between the first and second dependent variables, and `2_2` is the coefficient on the ARCH term for the conditional variance of the second dependent variable.

◁

▷ Example 2: Model with covariates that differ by equation

We improve the previous example by removing the insignificant parameters from the model:

```
. mgarch dvech (D.bond = LD.bond LD.tbill, noconstant)
> (D.tbill = LD.tbill, noconstant), arch(1)

Getting starting values
(setting technique to bhhh)
Iteration 0:  Log likelihood =  3566.8824
Iteration 1:  Log likelihood =  3701.6181
Iteration 2:  Log likelihood =  3952.8048
Iteration 3:  Log likelihood =  4076.5164
Iteration 4:  Log likelihood =  4166.6842
Iteration 5:  Log likelihood =  4180.2998
Iteration 6:  Log likelihood =  4182.4545
Iteration 7:  Log likelihood =  4182.9563
(switching technique to nr)
Iteration 8:  Log likelihood =  4183.0293
Iteration 9:  Log likelihood =  4183.1112
Iteration 10: Log likelihood =  4183.1113

Estimating parameters
(setting technique to bhhh)
Iteration 0:  Log likelihood =  4183.1113
Iteration 1:  Log likelihood =  4202.0304
Iteration 2:  Log likelihood =  4210.2929
Iteration 3:  Log likelihood =  4215.7798
Iteration 4:  Log likelihood =  4217.7755
(switching technique to nr)
Iteration 5:  Log likelihood =  4219.0078
Iteration 6:  Log likelihood =  4221.4197
Iteration 7:  Log likelihood =   4221.433
Iteration 8:  Log likelihood =   4221.433

Diagonal vech MGARCH model
Sample: 3 thru 2456                                   Number of obs =   2,454
Distribution: Gaussian                                Wald chi2(3)  = 1197.76
Log likelihood = 4221.433                             Prob > chi2   =  0.0000
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
D.bond						
bond						
LD.	.2941649	.0234734	12.53	0.000	.2481579	.3401718
tbill						
LD.	.0953158	.0098077	9.72	0.000	.076093	.1145386
D.tbill						
tbill						
LD.	.4385945	.0136672	32.09	0.000	.4118072	.4653817
/Sigma0						
1_1	.0048922	.0002005	24.40	0.000	.0044993	.0052851
2_1	.0040949	.0002394	17.10	0.000	.0036256	.0045641
2_2	.0115043	.0005184	22.19	0.000	.0104883	.0125203
L.ARCH						
1_1	.4519233	.045671	9.90	0.000	.3624099	.5414368
2_1	.2515474	.0366701	6.86	0.000	.1796752	.3234195
2_2	.8437212	.0600839	14.04	0.000	.7259589	.9614836

We specified each equation separately to remove the insignificant parameters. All the parameter estimates are statistically significant.

◁

▷ Example 3: Model with constraints

Here we analyze some fictional weekly data on the percentages of bad widgets found in the factories of Acme Inc. and Anvil Inc. We model the levels as a first-order autoregressive process. We believe that the adaptive management style in these companies causes the variances to follow a diagonal vech MGARCH process with one ARCH term and one GARCH term. Furthermore, these close competitors follow essentially the same process, so we impose the constraints that the ARCH coefficients are the same for the two companies and that the GARCH coefficients are also the same.

Imposing these constraints yields

```
. use https://www.stata-press.com/data/r18/acme
. constraint 1 [L.ARCH]1_1 = [L.ARCH]2_2
. constraint 2 [L.GARCH]1_1 = [L.GARCH]2_2
. mgarch dvech (acme = L.acme) (anvil = L.anvil), arch(1) garch(1)
> constraints(1 2)
Getting starting values
(setting technique to bhhh)
Iteration 0:  Log likelihood = -6087.0665  (not concave)
Iteration 1:  Log likelihood = -6022.2046
Iteration 2:  Log likelihood = -5986.6152
Iteration 3:  Log likelihood = -5976.5739
Iteration 4:  Log likelihood = -5974.4342
Iteration 5:  Log likelihood = -5974.4046
Iteration 6:  Log likelihood = -5974.4036
Iteration 7:  Log likelihood = -5974.4035
Estimating parameters
(setting technique to bhhh)
Iteration 0:  Log likelihood = -5974.4035
Iteration 1:  Log likelihood =  -5973.812
Iteration 2:  Log likelihood = -5973.8004
Iteration 3:  Log likelihood = -5973.7999
Iteration 4:  Log likelihood = -5973.7999
```

```
Diagonal vech MGARCH model
Sample: 1969w35 thru 1998w25                        Number of obs =  1,499
Distribution: Gaussian                              Wald chi2(2)  = 272.47
Log likelihood = -5973.8                            Prob > chi2   = 0.0000
 ( 1)  [L.ARCH]1_1 - [L.ARCH]2_2 = 0
 ( 2)  [L.GARCH]1_1 - [L.GARCH]2_2 = 0
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
acme						
acme						
L1.	.3365278	.0255134	13.19	0.000	.2865225	.3865331
_cons	1.124611	.060085	18.72	0.000	1.006847	1.242376
anvil						
anvil						
L1.	.3151955	.0263287	11.97	0.000	.2635922	.3667988
_cons	1.215786	.0642052	18.94	0.000	1.089947	1.341626
/Sigma0						
1_1	1.889237	.2168733	8.71	0.000	1.464173	2.314301
2_1	.4599576	.1139843	4.04	0.000	.2365525	.6833626
2_2	2.063113	.2454633	8.40	0.000	1.582014	2.544213
L.ARCH						
1_1	.2813443	.0299124	9.41	0.000	.222717	.3399716
2_1	.181877	.0335393	5.42	0.000	.1161412	.2476128
2_2	.2813443	.0299124	9.41	0.000	.222717	.3399716
L.GARCH						
1_1	.1487581	.0697531	2.13	0.033	.0120445	.2854716
2_1	.085404	.1446524	0.59	0.555	-.1981094	.3689175
2_2	.1487581	.0697531	2.13	0.033	.0120445	.2854716

We could test our constraints by fitting the unconstrained model and performing either a Wald or a likelihood-ratio test. The results indicate that we might further restrict the time-invariant components of the conditional variances to be the same across companies.

◁

▷ Example 4: Model with a GARCH term

Some models of financial data include no covariates or constant terms. For example, in modeling fictional data on the stock returns of Acme Inc. and Anvil Inc., we found it best not to include any covariates or constant terms. We include two ARCH terms and one GARCH term to model the conditional variances.

```
. use https://www.stata-press.com/data/r18/aacmer
. mgarch dvech (acme anvil = , noconstant), arch(1/2) garch(1)
Getting starting values
(setting technique to bhhh)
Iteration 0:  Log likelihood = -18417.243  (not concave)
Iteration 1:  Log likelihood = -18215.005
Iteration 2:  Log likelihood = -18199.691
Iteration 3:  Log likelihood = -18136.699
Iteration 4:  Log likelihood = -18084.256
Iteration 5:  Log likelihood = -17993.662
Iteration 6:  Log likelihood =   -17731.1
Iteration 7:  Log likelihood = -17629.505
(switching technique to nr)
Iteration 8:  Log likelihood = -17548.172
Iteration 9:  Log likelihood = -17544.987
Iteration 10: Log likelihood = -17544.937
Iteration 11: Log likelihood = -17544.937
Estimating parameters
(setting technique to bhhh)
Iteration 0:  Log likelihood = -17544.937
Iteration 1:  Log likelihood = -17544.937
Diagonal vech MGARCH model
Sample: 1 thru 5000                                   Number of obs = 5,000
Distribution: Gaussian                                Wald chi2(.)  =     .
Log likelihood = -17544.94                            Prob > chi2   =     .
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
/Sigma0						
1_1	1.026283	.0823348	12.46	0.000	.8649096	1.187656
2_1	.4300997	.0590294	7.29	0.000	.3144042	.5457952
2_2	1.019753	.0837146	12.18	0.000	.8556751	1.18383
L.ARCH						
1_1	.2878739	.02157	13.35	0.000	.2455975	.3301504
2_1	.1036685	.0161446	6.42	0.000	.0720256	.1353114
2_2	.2034196	.019855	10.25	0.000	.1645044	.2423347
L2.ARCH						
1_1	.1837825	.0274555	6.69	0.000	.1299706	.2375943
2_1	.0884425	.02208	4.01	0.000	.0451665	.1317185
2_2	.2025718	.0272639	7.43	0.000	.1491355	.256008
L.GARCH						
1_1	.0782467	.053944	1.45	0.147	-.0274816	.183975
2_1	.2888104	.0818303	3.53	0.000	.1284261	.4491948
2_2	.201618	.0470584	4.28	0.000	.1093853	.2938508

The model test is omitted from the output, because there are no covariates in the model. The univariate tests indicate that the included parameters fit the data well. In [TS] **mgarch dvech postestimation**, we discuss prediction from models without covariates.

◁

Stored results

`mgarch dvech` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k)`	number of parameters
`e(k_extra)`	number of extra estimates added to `_b`
`e(k_eq)`	number of equations in `e(b)`
`e(k_dv)`	number of dependent variables
`e(df_m)`	model degrees of freedom
`e(ll)`	log likelihood
`e(chi2)`	χ^2
`e(p)`	p-value for model test
`e(estdf)`	`1` if distribution parameter was estimated, `0` otherwise
`e(usr)`	user-provided distribution parameter
`e(tmin)`	minimum time in sample
`e(tmax)`	maximum time in sample
`e(N_gaps)`	number of gaps
`e(rank)`	rank of `e(V)`
`e(ic)`	number of iterations
`e(rc)`	return code
`e(converged)`	`1` if converged, `0` otherwise

Macros

`e(cmd)`	`mgarch`
`e(model)`	`dvech`
`e(cmdline)`	command as typed
`e(depvar)`	names of dependent variables
`e(covariates)`	list of covariates
`e(dv_eqs)`	dependent variables with mean equations
`e(indeps)`	independent variables in each equation
`e(tvar)`	time variable
`e(title)`	title in estimation output
`e(chi2type)`	`Wald`; type of model χ^2 test
`e(vce)`	*vcetype* specified in `vce()`
`e(vcetype)`	title used to label Std. err.
`e(tmins)`	formatted minimum time
`e(tmaxs)`	formatted maximum time
`e(dist)`	distribution for error term: `gaussian` or `t`
`e(arch)`	specified ARCH terms
`e(garch)`	specified GARCH terms
`e(svtechnique)`	maximization technique(s) for starting values
`e(technique)`	maximization technique
`e(properties)`	`b V`
`e(estat_cmd)`	program used to implement `estat`
`e(predict)`	program used to implement `predict`
`e(marginsok)`	predictions allowed by `margins`
`e(marginsnotok)`	predictions disallowed by `margins`
`e(marginsdefault)`	default `predict()` specification for `margins`
`e(asbalanced)`	factor variables `fvset` as `asbalanced`
`e(asobserved)`	factor variables `fvset` as `asobserved`

Matrices

e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(hessian)	Hessian matrix
e(A)	estimates of A matrices
e(B)	estimates of B matrices
e(S)	estimates of Sigma0 matrix
e(Sigma)	Sigma hat
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance
e(pinfo)	parameter information, used by predict

Functions

e(sample)	marks estimation sample

In addition to the above, the following is stored in r():

Matrices

r(table)	matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in r() are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Recall that the diagonal vech MGARCH model can be written as

$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \boldsymbol{\epsilon}_t$$

$$\boldsymbol{\epsilon}_t = \mathbf{H}_t^{1/2}\boldsymbol{\nu}_t$$

$$\mathbf{H}_t = \mathbf{S} + \sum_{i=1}^{p} \mathbf{A}_i \odot \boldsymbol{\epsilon}_{t-i}\boldsymbol{\epsilon}_{t-i}' + \sum_{j=1}^{q} \mathbf{B}_j \odot \mathbf{H}_{t-j}$$

where

$\mathbf{y}_t$ is an $m \times 1$ vector of dependent variables;

$\mathbf{C}$ is an $m \times k$ matrix of parameters;

$\mathbf{x}_t$ is a $k \times 1$ vector of independent variables, which may contain lags of $\mathbf{y}_t$;

$\mathbf{H}_t^{1/2}$ is the Cholesky factor of the time-varying conditional covariance matrix $\mathbf{H}_t$;

$\boldsymbol{\nu}_t$ is an $m \times 1$ vector of normal, independent, and identically distributed innovations;

$\mathbf{S}$ is an $m \times m$ symmetric matrix of parameters;

each $\mathbf{A}_i$ is an $m \times m$ symmetric matrix of parameters;

$\odot$ is the elementwise or Hadamard product; and

each $\mathbf{B}_j$ is an $m \times m$ symmetric matrix of parameters.

mgarch dvech estimates the parameters by maximum likelihood. The log-likelihood function based on the multivariate normal distribution for observation t is

$$l_t = -0.5m\log(2\pi) - 0.5\log\left\{\det\left(\mathbf{H}_t\right)\right\} - 0.5\boldsymbol{\epsilon}_t\mathbf{H}_t^{-1}\boldsymbol{\epsilon}_t'$$

where $\boldsymbol{\epsilon}_t = \mathbf{y}_t - \mathbf{C}\mathbf{x}_t$. The log-likelihood function is $\sum_{t=1}^{T} l_t$.

If we assume that $\boldsymbol{\nu}_t$ follow a multivariate t distribution with degrees of freedom (df) greater than 2, then the log-likelihood function for observation t is

$$l_t = \log\Gamma\left(\frac{\mathrm{df}+m}{2}\right) - \log\Gamma\left(\frac{\mathrm{df}}{2}\right) - \frac{m}{2}\log\{(\mathrm{df}-2)\pi\} - 0.5\log\{\det(\mathbf{H}_t)\} - \frac{\mathrm{df}+m}{2}\log\left(1+\frac{\boldsymbol{\epsilon}_t\mathbf{H}_t^{-1}\boldsymbol{\epsilon}_t'}{\mathrm{df}-2}\right)$$

`mgarch dvech` ensures that $\mathbf{H}_t$ is positive definite for each t.

By default, `mgarch dvech` performs an iterative search for starting values. `mgarch dvech` estimates starting values for $\mathbf{C}$ by seemingly unrelated regression, uses these estimates to compute residuals $\widehat{\boldsymbol{\epsilon}}_t$, plugs $\widehat{\boldsymbol{\epsilon}}_t$ into the above log-likelihood function, and optimizes this log-likelihood function over the parameters in $\mathbf{H}_t$. This starting-value method plugs in consistent estimates of the parameters for the conditional means of the dependent variables and then iteratively searches for the variance parameters that maximize the log-likelihood function. Lütkepohl (2005, chap. 16) discusses this method as an estimator for the variance parameters.

GARCH estimators require initial values that can be plugged in for $\boldsymbol{\epsilon}_{t-i}\boldsymbol{\epsilon}_{t-i}'$ and $\mathbf{H}_{t-j}$ when $t-i<1$ and $t-j<1$. `mgarch dvech` substitutes an estimator of the unconditional covariance of the disturbances,

$$\widehat{\boldsymbol{\Sigma}} = T^{-1}\sum_{t=1}^{T}\widehat{\boldsymbol{\epsilon}}_t\widehat{\boldsymbol{\epsilon}}_t' \tag{4}$$

for $\boldsymbol{\epsilon}_{t-i}\boldsymbol{\epsilon}_{t-i}'$ when $t-i<1$ and for $\mathbf{H}_{t-j}$ when $t-j<1$, where $\widehat{\boldsymbol{\epsilon}}_t$ is the vector of residuals calculated using the estimated parameters.

`mgarch dvech` uses analytic first and second derivatives in maximizing the log-likelihood function based on the multivariate normal distribution. `mgarch dvech` uses numerical derivatives in maximizing the log-likelihood function based on the multivariate t distribution.

References

Bollerslev, T., R. F. Engle, and J. M. Wooldridge. 1988. A capital asset pricing model with time-varying covariances. *Journal of Political Economy* 96: 116–131. https://doi.org/10.1086/261527.

Lütkepohl, H. 2005. *New Introduction to Multiple Time Series Analysis*. New York: Springer.

Also see

[TS] **mgarch dvech postestimation** — Postestimation tools for mgarch dvech

[TS] **arch** — Autoregressive conditional heteroskedasticity (ARCH) family of estimators

[TS] **mgarch** — Multivariate GARCH models

[TS] **tsset** — Declare data to be time-series data

[TS] **var** — Vector autoregressive models+

[U] **20 Estimation and postestimation commands**