A Quick Guide to GPE/GAUSS

GPE (GAUSS Programming for Econometricians and Financial Analysts) is a package of econometric procedures written in GAUSS, and this Guide is about GAUSS programming for econometric analysis and applications using GPE.

With many econometric procedures controllable by a few groups of global variables, the GPE package covers most econometric computations including single linear equation estimation and prediction, simultaneous linear equations system, nonlinear models, and time series analysis.

Installation Notes

Before using the GPE econometric package, it must be properly installed with your GAUSS program. Install GPE according to the instruction given with the distribution disk. Make sure that the version number of GPE matches with that of your GAUSS program. Following the completion of GPE installation, the compiled GPE program named GPE2.GCG should reside in the GAUSS directory. In addition, a GPE sub-directory is created and stores all the lesson programs and data files. GPE is the working directory for all the empirical lessons. In particular, you will see the following files in the GAUSS directory:

GPE2.GCG	GPE compiled program, version 2
GPE*.TXT	Data files for example lessons, * = data file name
GPE\LESSON?.?	Lesson programs, ?.? = lesson number

How to Use GPE

To use the GPE econometric package in your program, the first thing you have to do is to include the following statement in the very beginning of your program:

```
use gpe2;
```

This will load the GPE package (version 2) for econometric analysis and applications. Note that USE can appear only once at the top of a program.

GPE consists of four main procedures (ESTIMATE, FORECAST, OPTIMIZE and RESET) along with global control variables that modify the econometric routines. The procedure ESTIMATE computes linear and generalized linear regressions, OPTIMIZE estimates nonlinear models using optimization methods, and FORECAST performs least squares prediction for linear and generalized linear models. The RESET procedure initializes global

control variables to their default values. For backward compatibility with earlier versions of GPE2, ESTIMATE can be used for nonlinear model estimation as well.

For reference purposes, consider the following general regression equation:

$F(Z, \beta) = \varepsilon$

where Z is the data matrix of variables and b is the vector of parameters, which define the functional form F. ε is the error structure of the model. Z can be further decomposed as Z = [Y, X] with Y denoting the endogenous (dependent) variables and X the predetermined (independent) variables. If Y consists of more than one column, it is a system of linear equations. For a classical regression model, $Y = f(X, \beta) + \varepsilon$ or $F(Z, \beta) = Y - f(X, \beta) = \varepsilon$. The simplest case of single linear regression equation is written as:

$Y = X\beta + \varepsilon$

where Y is the left-hand-side (LHS) or dependent variable, and X is the right-hand-side (RHS) explanatory or independent variables. β is the vector of estimated parameters, and ε is the vector of estimated residuals.

In your program, to estimate the model, the following statement is used (notice that GAUSS program is case insensitive by default):

```
call estimate(y,x);
```

Similarly, to forecast the model, the following is used:

```
call forecast(y,x);
```

To estimate a nonlinear model defined by a error or residual function $F(Z, \beta) = \varepsilon$, the procedure OPTIMIZE is called as follows:

```
call optimize(&F,Z,b);
```

where &F points to the function defined, which takes data matrix Z and a vector b of initial values of parameters. Alternatively, ESTIMATE can be used with the initial values of parameters given as an input control variable defined below. The procedure OPTIMIZE is more flexible in estimating the nonlinear model defined by a complicate nonlinear objective function.

GPE Control Variables

There are two types of global control variables in GPE: input control variables and output control variables. Global input variables control the execution behavior of the called

procedure. For example, they can modify ESTIMATE and FORECAST to use linear restrictions, weighted variables, instrumental variables, lagged dependent and independent variables, etc.

Output global variables are the results of calling the main procedures ESTIMATE, OPTIMIZE, and FORECAST. They can be assigned to new variables for further analysis. Depending on the input global variables used which control the econometric routines, not all the output global variables will be available. The name of an input control variable starts with a single underscore (for example, _b), while an output control variable starts with a double underscore (for example, __b).

Global control variables must be initialized before calling one of the main econometric routines: ESTIMATE, OPTIMIZE, or FORECAST. To initialize all the global control variables, call RESET will do it. That is,

```
call reset;
```

GPE Input Control Variables

Three categories of input control variables are listed below: general-purpose input control variables, ESTIMATE and OPTIMIZE input control variables, and FORECAST input control variables. Unless otherwise specified, setting each variable to 1 (that is, True or Yes) activates or turns on the optional behavior specified. If the variable is not defined or specified, then its default value is assumed.

General Purpose Input Control Variables

Variable	**Description**
_cmplx	Complex number computation. _cmplx = 0 (default): Do not allow for complex number computation, therefore negative argument for LN, LOG, and SQRT is not permitted; _cmplx =1: Allow for complex number computation.
_legend	When a graph is requested (_rplot>0, see below), _legend = 1 (default): Show legends for graph plots; _legend = 0: No legends will be shown.
_pause	Pause the output before displaying graphs. _pause = 0 (default): No waiting prompt; _pause = 1: Wait for a keystroke to display graphs.
_print	Controls screen output. _print = 1 (default): Direct full output to screen; _print = 0: Direct partial output to screen. Verbose iteration outputs from

a nonlinear or iterative model are suppressed;
_print = -1: Suppress all screen output. Suppressing the screen output will force _rplot = 0 and _fplot = 0 (see below), but it will have no effect on sending output to a file or printer if requested.

ESTIMATE and OPTIMIZE Input Control Variables

Variable	**Description**
_acf	Specify the number of lags for computing autocorrelation and partial autocorrelation coefficients from the estimated regression residuals. Useful for testing the ARMA error structure. Display and plot the functions if _rplot > 0 (see below). In addition, standard errors of coefficients and Box-Pierece and Ljung-Box portmanteau test statistics are presented up to the number of lags specified in _acf. For example, 12 lags of autocorrelation and partial autocorrelation functions are requested by setting: _acf = 12; _acf = 0; (default) As an option for computing autocorrelation coefficients and the associated standard errors using regression method, the second element of the vector _acf may be set to a positive value, with the first element indicating the number of lags requested. For example: _acf = {12,1};
_acf2	Same as _acf except that the autocorrelation and partial autocorrelation coefficients are computed from the squared standardized residuals. Useful for testing the GARCH error structure. _acf2 = 0; (default)
_ar	Specify the order of an autoregressive (AR) error structure. If an additional moving average process is desired for anautoregressive moving average ARMA structure, use the variable _arma instead (see below). Optional initial values of the autocorrelation coefficients may be appended at the end of _ar for estimation. Providing the initial values is useful for starting a search from non-zero values of autocorrelation coefficients. Missing order is specified by putting a missing character (.) for the missing element in the vector of initial values. For example: _ar = 1; _ar = {1, 0.5}; (with initial value of AR(1) parameter) _ar = {2, . , 0}; (AR(2) with the first order parameter missing) _ar = 0; (default)
_arma	Specify the orders of an autoregressive moving average (ARMA) error

structure. It is a column vector consisting of at least two elements. The first element denotes the order of autoregressive portion of the ARMA process, while the second element is the order of moving average portion. If only the autoregressive portion is given, it is exactly the AR model (see _ar above). The model is estimated using maximum likelihood method conditional to the initialization of pre-sample series, which is the sample mean of the error series. Optional initial values of the autoregressive and moving average coefficients may be appended at the end of _arma for estimation. Giving the initial values is useful for starting a search from non-zero values of ARMA coefficients. Missing order is specified by putting a missing character (.) for the missing element in the vector of initial values. For example:
_arma = {1, 0}; (this is identical to: _ar = 1;)
_arma = {0, 1};
_arma = {1, 1, 0.5, -0.5}; (initial values of ARMA(1,1) parameters)
_arma = {2, 1, . , 0, 0}; (ARMA(2,1) with missing AR(1))
_arma = {0, 0}; (default)

`_b` A column vector of initial parameter values for nonlinear model estimation.

`_begin` Specify the starting observation number for estimation.
_begin = 1; (default)

`_bjtest` Bera-Jarque test for residual normality.
_bjtest = 0 (default): Skip the test;
_bjtest = 1: Perform the test.

`_bgtest` Breusch-Godfrey test for higher-order autocorrelation.
_bgtest = 0 (default): Skip the test;
_bgtest = p (>0): Perform the test for autocorrelation up to the p-th order. The number p (>0) is the highest order tested.

`_bptest` Breusch-Pagan and White tests for heteroscedasticity.
_bptest = 0 (default): Skip the test;
_bptest = 1: Perform Breusch-Pagan and White tests for general heteroscedasticity. For the Breusch-Pagan test, all explanatory variables including constant term (i.e., X) are the RHS variables of the auxiliary test regression. For the White test, all explanatory variables and their squares and cross product including constant term are the RHS variables of the auxilary test regression. Alternatively, _bptest can be set to a data matrix (a subset of X or including other variables) in place of X for use with the test for heteroscedasticity.

`_const` Specify a constant term for a regression model.

_const = 1 (default): Constant term is added in the regression;
_const = 0: No constant is added.

For a system model, this is a column vector of 0 (no constant) or 1 (with constant) associated with each equation.

_conv Convergence criteria for nonlinear model estimation.
_conv = 0 (default): Convergence in function value and solution;
_conv = 1: Convergence in function value, solution, and zero gradient. All convergence criteria are checked relative to the tolerance level _tol (see below).

_corr Compute condition number of explanatory variables and correlation matrix of dependent and explanatory variables, useful for multicollinearity analysis.
_corr = 0 (default): Do not compute the statistics;
_corr = 1: Compute and show the statistics.

_deriv For a nonlinear optimization problem, if the first analytical derivative function is defined and used, _deriv is set to the location (address) of the function. If the second analytical derivative function is also given, its location is vertically concanated with the first. If analytical derivative functions are not used, the numerical derivatives are computed instead. That is,
_deriv = 0; (default)

_dlags A scalar or a 2x1 column vector to specify the use of lagged dependent variables. As a scalar, it is the order of the regular lagged dependent variables in use. As a 2x1 column vector, a seasonal lagged dependent variables model is identified with order _dlags[1] and seasonal span _dlags[2] (require _dlags[2]>0). Normally, _dlags[2] = 4 for a model with quarterly data series, while _dlags[2] = 12 for the monthly case. _dlags[1] or the scalar _dlags is always the order number. For a pure (regular or seasonal) lagged dependent variables model, set RHS variable X = 0 in calling ESTIMATE procedure and specify the column vector _dlags accordingly. For example:
_dlags = q; (or equivalently, _dlags = {q,1};)
_dlags = {q,s};
Where q is the order of autocorrelation and s is the seasonal span.
_dlags = 0; (default)

For a system model, _dlags is a g×g matrix with the value of its entry indicating the number of lags for each endogenous variable (column) in each equation (row). A zero ij-element of _dlags signifies that no lag is used for the j-th variable in the i-th equation. Here, g is the number of

endogenous variables or equations.

_drop — Drop the first few observations for model estimation. Depending on the method of estimation, initial unusable observations may be dropped automatically.
_drop =1: Drop the first observation or the first seasonal span of observations for AR model estimation;
_drop =0 (default): Keep the first observation or the first seasonal span of observations for AR model estimation with appropriate data transformation.

_ebtest — Engle-Bollerslev test for higher-order autoregressive conditional heteroscedasticity (ARCH).
_ebtest = 0 (default): Skip the test;
_ebtest = q (>0) :Perform the test for ARCH structure up to the q-th order. The number q (>0) is the highest order tested.

_end — Specify the ending observation number for estimation.
_end = rows(y) (default).

_eq — Specify the stochastic equation specification matrix for system model estimation. This is a g×(gs+ks) matrix with elements -1, 0, and 1 arranged in accordance with the order of endogenous variables followed by the predetermined variables. Note that g is the number of stochastic equations, gs is the number of endogenous variables (gs>=g), while ks is the number of predetermined variables. In the stochastic equation specification matrix _eq, an element -1 indicates the LHS endogenous variable. Only one -1 entry is allowed in each equation. An element 1 indicates the use of an endogenous and/or a predetermined variable on the RHS of an equation. An element 0 indicates the corresponding unused variable. If _eq is not specified, or _eq=0 by default, a restricted seemingly unrelated system with common parameters across equations is assumed. That is, g=gs and -1 in the gs diagonals and 1 in the next gs×ks predetermined variables portion of the matrix. If _eq is a scalar with value 1 or 2, then an unrestricted seemingly unrelated system is assumed. If _eq=1, the RHS variables are grouped by variables in the order of equations. That is, there are (ks/gs) blocks of identity (gs×gs) matrices concatenated horizontally, which occupy the gs×ks predetermined variables portion of the matrix. If _eq=2, the RHS variables are grouped by equations in the order of variables. That is, there are gs blocks of 1× (ks/gs) row vector of ones spanning over the gs×ks predetermined variables portion of the matrix along the diagonal. Seemingly unrelated system with other forms of arrangements of RHS variables must be specified with the matrix form of _eq. Normally

constant term is not needed in the equation specification, and it is automatically included in each equation.

_garch Specify the orders of a generalized autoregressive conditonal heteroscedasticity (GARCH) error structure. It is a column vector consisting of at least two elements. The first element denotes the order of autoregressive (variances) portion of GARCH process, while the second element is the order of moving average (squared errors) portion. The model is estimated using maximum likelihood method conditional to the initialization of pre-sample series, which is the sample variance of the error series. The optional initial value of GARCH coefficients may be appended at the end of _garch for estimation. Be reminded that there is always a constant for the GARCH process. The constant is the last term of GARCH parameters. Giving the initial values is useful for starting a search from non-zero values of GARCH coefficients. Missing order is specified by putting a missing character (.) for the missing element in the vector of initial values. For example:
_garch = {1, 0};
_garch = {0, 1};
_garch = {1, 1, 0.5, 0.5, 0.5}; (with initial values of GARCH(1,1))
_garch = {0, 0}; (default)

_garchx Specify additional variables included in the GARCH variance equation (see _garch above). This may be a data matrix of multiple variables. The variables must be defined with the same number of rows or observations as that of the regression residuals.

_garcha Add asymmetry in the GARCH variance equation according to GJR specification. The asymmetric response is estimated for the negative errors with the same order specified for the moving average (squared errors) portion of the GARCH variance equation (see _garch above).
_garcha = 0 (default): No asymmetry;
_garcha = 1: Compute asymmetric response.
Initial values of the asymmetry parameters may be appended.

_garchllf Allow non-normal distribution for GARCH model specification.
_garchllf = 1: Generalized exponential distribution (GED);
_garchllf = 2: Student-t distribution;
_garchllf = 3: Skew Student-t distribution (Hansen, 1994);
_garchllf = 4: Skew Student-t distribution (Azzalini, 1985);
_garchllf = 0 (default): Normal distribution.
Initial values of the distribution parameters may be appended.

_hacv Compute heteroscedasticity and autocorrelation-consistent variance-covariance matrix and perform adjustment to standard error and t-ratio

of estimated coefficients accordingly. This may be a column vector up to 3 elements.
_hacv = 0 (default): No adjustment;
_hacv = 1: Compute heteroscedasticity consistent variance-covariance matrix;
_hacv = {0,p}: Compute p-th order autocorrelation-consistent variance-covariance matrix with declining weights of autocovariances (Newey-West estimators), p=1,2,...;
_hacv = {1,p}: Compute heteroscedasticity and p-th order autocorrelation consistent variance-covariance matrix with declining weights of autocovariances (Newey-West estimators), p=1,2,....;
_hacv = {1,p,1}: Compute heteroscedasticity and p-th order autocorrelation-consistent variance-covariance matrix with the equal weighted autocovariances, p=1,2,....; Therefore, _hacv = {0,p}is the same as _hacv = {0,p,0} and _hacv = {1,p}is the same as _hacv = {1,p,0}.

Note: If _hacv is used in conjunction with the instrumental variable estimation (see, _ivar below) in setting the number of iterations (see, _iter below) to achieve the convergence of estimated parameters, this is essentially the Generalized Method of Moments (GMM). _hacv is meaningful only when there is potential misspecification problem of autoregressive and/or heteroscedastic error structure.

_hacv may be used in conjunction with the system model estimation. If the method of 2SLS (see _method below) is requested with non-zero _hacv, then two stage GMM estimation is performed. Similarly, if the method of 3SLS is requested with non-zero _hacv, then three stage GMM estimation is performed. However, the computed variance-covariance matrix for the system may become non-positive definite due to excess numerical rounding errors or the improper autocovariance structures specified.

`_id` Specify the identity equation specification matrix for a system model. Similar size and setup as _eq (see above) except that its entries can be any value as required. If _id is not specified, or _id=0 by default, there is no identity. Note: gs=rows(_eq|_id) to ensure the system compatibility.

`_iter` Maximum number of iterations requested for iterative or nonlinear model estimation.
_iter = 1; (default)

`_ivar` Instrumental variable estimation requested with instrumental variables specified in matrix _ivar. If _dlags[1] > 0, _ivar may be given with a

positive scalar (i.e. _ivar = 1) and perform instrumental variable estimation with the internal instrumental variables obtained from the explanatory variables and their lags. If the matrix form of _ivar is specified, the external instrumental variables are used for that cols(_ivar) >= cols(X) + _const and rows(_ivar) >= rows(X). Constant term is automatically included in _ivar.
_ivar = 0; (default)

For a system model, external instrumental variable estimation may be requested with the instrumental variables specified in matrix _ivar. The data matrix _ivar will be combined with all predetermined variables to form the basis for instrumental variable estimation.

_ivar may be used together with _iter and _hacv (see above) to produce the GMM estimation.

`_jacob` Controls the use of Jacobians in deriving the log-likelihood function from the residuals. Since a correct log-likelihood function may include non-vanishing log-Jacobian terms, the Jacobian transformation which is a function of the estimated parameters should be defined. If the analytical Jacobian function is used, _jacob is set to the location (address) of the function. If you do not wish to write out the Jacobian analytically, you may set _jacob = 1. Then the numerical Jacobian is computed for each sample observation, which is usually a time consuming process. In case of requesting numerical Jacobians, the first column of the data matrix used to define the residuals must be the dependent variable Y (recall that $J(Z,\varepsilon) = |\partial\varepsilon/\partial Y|$ and $Z = [Y,X]$).
_jacob=0 (default): do not use Jacobians.

`_log` Apply log transformation to the data series. Only the positive data series can be log transformed. _log may be a scalar or a vector of 1 (log transformed) or 0 (not transformed) for the variables in the order of dependent and independent variables as appeared in the ESTIMATE procedure. If the length of _log is shorter than the list of variables in the regression, the rest of variables are not transformed. If the dependent variable is log transformed, the model is assumed to be a log-normal linear model. The advantage of using _log instead of hard coding the variable transformation in the program is that the exact log-likelihood is computed when the dependent variable is logged or _log[1] = 1. For example:
_log = 1; (dependent variable is log transformed)
_log = {0, 1, 1}; (only the first two independent variables are log transformed)
_log = {1, 1, 0, 1}; (dependent variable and the first two and forth independent variables are log transformed; the third one is left alone; if

there are more than four independent variables used in the regression, they are not transformed)
_log = 0; (default, do not apply log transformation)

When calling FORECAST with _log[1] set to 1 in the previous ESTIMATE procedure, the unlogged series of forecasts is reported in addition to the log transformed series.

_ma

Specify the order of a moving average (MA) error structure. If an additional autoregressive process is desired for an autoregressive moving average ARMA structure, use the variable _arma instead (see above). Optional initial values of the moving average coefficients may be appended at the end of _ma for estimation. Providing the initial values is useful for starting a search from non-zero values of moving average coefficients. Missing order is specified by putting a missing character (.) for the missing element in the vector of initial values. For example:
_ma = 1;
_ma = {1, 0.5}; (with initial value of MA(1) parameter)
_ma = {2, . , 0}; (MA(2) with the first order parameter missing)
_ma = 0; (default)

_method

Specify the estimation method for an AR model.
_method = 0 (default): Cochrane-Orcutt iterative LS method;
_method = {0,1}: Cochrane-Orcutt iterative ML method;
method = 1: Beach-MacKinnon iterative ML method (for _ar=1 only, and _drop=0 is in effect);
_method = 2 or {2,0}: Hildreth-Lu grid search LS method (for _ar=1 only);
_method = {2,1}: Hildreth-Lu grid search ML method (for _ar=1 only).
Note: higher AR order (_ar>1) can only use _method = 0 or method = {0,1}.

Specify the estimation method for a system model.
_method = 0 (default): Ordinary least squares (biased);
_method = 1: Limited information maximum likelihood;
_method = 2: Two-stage least squares;
_method = 3: Three-stage least squares;
_method = 4: Full information maximum likelihood.
Note: LIML and FIML are not true nonlinear maximum likelihood estimation. Instead they are types of instrumental variables estimation. Three variants of the FIML method are available:
_method = {4,0} (or 4): FIML instrumental variable method;
_method = {4,1}: FIML linearized method;
_method = {4,2}: FIML Newton method.

Specify the estimation method for a nonlinear model (including a linear model with nonlinear error structure such as ARMA and GARCH).
_method = 0 (default): Steep-ascent or decent method for mathematical optimization; Gauss-Newton method for nonlinear least squares estimation; Berndt-Hall-Hall_Hausman (BHHH) method for maximum likelihood estimation;
_method = 1: Quasi-Newton BFGS update method;
_method = 2: Quasi-Newton DFP update method;
_method = 3: Greenstadt method;
_method = 4: Newton-Raphson method;
_method = 5: Quadratic hill-climbing method;
_method = 6: Modified quadratic hill-climbing method.

`_names` Specify a vector of character names for variables (linear model) or parameters (non-linear model) as appeared in a regression equation.

`_nlopt` Specify a nonlinear optimization problem.
_nlopt = 0 (default): Mathematical minimization of a scalar-valued function or nonlinear least squares estimation based on a vector-valued component error function; For the later case, the minimization is performed on the sum of squares of the vector-valued error function;
_nlopt = 1: Mathematical maximization of a scalar-valued function or maximum likelihood estimation based on a vector-valued component error function. For the later case, the maximization is performed on the sum of normal log-likelihood of the vector-valued component error function;
_nlopt = 2 Maximum likelihood estimation based on a vector-valued component log-likelihood function. The maximization is performed on the sum of the vector-valued component log-likelihood function;
_nlopt = -1 Minimum distance estimation based on a vector-valued component distance function (e.g., squares of the vector-valued component error function). The minimization is performed on the sum of the vector-valued component distance function.

`_pdl` Specify a polynomial distributed lag model if _pdl is defined as a rows(_pdl)x3 matrix. Each row of _pdl consists three elements: {q p r} where q = lags, p = orders, and r = endpoint restrictions: -1 (beginning), 1 (ending), 2 (both), and 0 (no restriction), for each RHS variable. Requires rows(_pdl) = cols(X), and cols(_pdl) = 3:
_pdl = 0; (default)

`_restart` Number of times to restart estimation for iterative or nonlinear models when function value does not improve. Maximum value of _restart is 10.

_restart = 0; (default)

_restr Perform restricted least squares estimation with the linear restrictions defined in accordance with the form: Rb = q, or [R1 R0][b1 b0]' = q, where b1 is the vector of slope coefficients and R1 is the restriction matrix corresponds to b1. Similarly, b0 is the intercept and R0 corresponds to b0. q is the vector of restricted values. Linear restrictions involving intercept should be specified in conjunction with _const = 0. If _restr is specified, then _restr = [R1 q]. Requires rows(_restr) = number of restrictions, and cols(_restr) = cols(X).
_restr = 0; (default)

For a system model, restrictions in the matrix R are stacked horizontally in accordance with the equations, while the vertical rows indicate the number of restrictions imposed. Own or cross equation restrictions can be coded easily. In general restrictions on the constant terms are not required.

_rlist List regression residual series.
_rlist = 0 (default): Skip listing the series;
_rlist = 1: List observed, predicted, and least squares residual series;
_rlist = 2: In addition to listing least squares residual series, studentized residuals and leverage information are provided. Useful for checking influential observations and outliers.

_rplot Plot regression residual series.
_rplot = 0 (default): No plots;
_rplot = 1: Plot residuals only;
_rplot = 2: Plot both observed and predicted, and residuals.
Also for plotting autocorrelation and partial autocorrelation functions if requested (see _acf above), a positive value of _rplot is needed.

For a nonlinear model, residual plot is meaningful only for the cases specified with error component functions (i.e., _nlopt=0 or 1). In defining the error component function, the dependent variable must be the first column of the data matrix and it has not been transformed within the definition of error component function.

_rstat Report regression residual statistics.
_rstat = 0 (default): Do not report the statistics;
_rstat =1: Report residual statistics, including DW, DH whenever appropriate.

For a nonlinear model, residual statistics are meaningful only for the

cases specified with error component functions (i.e., _nlopt=0 or 1). In defining the error component function, the dependent variable must be the first column of the data matrix and it has not been transformed within the definition of error component function.

_rtest

Hypothesis testings based on regression residual series. This is a 4-element row vector:
_rtest[1] is the same as _bjtest (see above) for Bera-Jarque test for residual normality;
_rtest[2] is the same as _bptest (see above) for Breusch-Pagan test for heteroscedasticity;
_rtest[3] is the same as _bgtest (see above) for Breusch-Godfrey test for higher-order autocorrelation;
_rtest[4] is the same as _ebtest (see above) for Engle-Bollerslev test for higher-order autoregressive conditional heteroscedasticity.
_rtest = 0 (default): No tests are performed.

_step

Specify step size of line search method for iterative or nonlinear model estimation.
_step = 0 (default): Cut back (half) step size is used;
_step = 1: Quadratic step size is used.

_tol

Set the convergence tolerance level for iterative or nonlinear model estimation.
_tol = 0.001; (default)

_vcov

Report the estimated variance-covariance matrix.
_vcov = 0 (default): Do not report the variance-covariance matrix;
_vcov = 1: Report variance-covariance matrix and correlation matrix of the estimated coefficients.

For a nonlinear model,
_vcov = 1: Variance-covariance matrix is derived from the method dependent approximated hessian (information matrix);
_vcov = 2: Variance-covariance matrix is derived from the estimated hessian.
_vcov = 3: Robust variance-covariance matrix, if available, is derived from the maximum likelihood estimation of component error or log-likelihood function.

_weight

Perform weighted least squares estimation with the weighting variable defined in _weight. _weight must be a column vector and rows(_weight) >= rows(X).
_weight = 0; (default)

FORECAST Input Control Variables

In addition to the ESTIMATE input variables which control the model specification (e.g., _ar, _arma, _dlags, _pdl, etc.), the following are the FORECAST input variables:

Variable	Description
_dynamic	Dynamic forecasts for lagged dependent variables model. _dynamic = 0 (default): Do not perform dynamic forecasts; _dynamic = 1: Perform dynamic forecasts. Dynamic forecast uses previous predicted lagged dependent variables.
_fbegin	Start of forecast observation number. _fbegin = _end +1; (default)
_fend	End of forecast observation number. _fend = rows(X); (default)
_fplot	Plots predicted or forecast series. _fplot = 0 (default): Do not plots the series; _fplot = 1: Plot predicted or forecast series.
_fstat	Computes ex-post forecast error statistics. _fstat = 0 (default): Do not compute the statistics. _fstat = 1: Compute and report the statistics.

Note: FORECAST is not available for nonlinear models.

GPE Output Control Variables

Output control variables are available after calling the procedure ESTIMATE, OPTIMIZE, or FORECAST. They may be used later in the program for further analysis. Depending on the input variables specified, not all the output variables will be available. Calling RESET assigns all output variables to zero. Each call to ESTIMATE, OPTIMIZE, or FORECAST assigns new values to output variables.

ESTIMATE Output Control Variables

Variable	Description
__a	Estimated coefficients of the autocorrelated error structure. Depending on the model specification, it may include AR or ARMA, and GARCH coefficients in that order.
__b	Estimated regression coefficients. For a simultaneous linear equations

model, see also __d and __pi.

__d	The structural form parameter matrix of a simultaneous linear equations system. See also __b and __pi.
__dh	Estimated Durbin-H statistic.
__dw	Estimated Durbin-Watson statistic.
__e	Estimated regression residuals; for nonlinear scalar-valued function optimization, this is the function value at the solution.
__g	Gradient vector of nonlinear objective function evaluated at the solution.
__h	Hessian matrix of nonlinear objective function evaluated at the solution.
__hat	Diagonal vector of Hat-matrix, $X(X'X)^{-1}X'$, or leverage.
__ll	Maximum log-likelihood function value.
__pi	The reduced form parameter matrix of a simultaneous linear equations system, useful for computing forecasts and multiplier analysis. See also __b and __d.
__r2	R-square (goodness of fit of the regression).
__r2a	Adjusted R-square.
__rss	Residual or error sums-of-squares.
__t	Estimated t-ratio for each of the regression parameters.
__v	Estimated regression variance.
__vb	Estimated variance-covariance matrix of the regression coefficients.
__ve	Estimated variances of the regression residuals.
__x	Data matrix of explanatory variables used in a linear model estimation.
__y	Data vector of dependent variable used in a linear model estimation.

FORECAST Output Control Variables

Variable	Description
__f	Predicted or forecast series.
__mape	Mean absolute percent of forecast errors
__mse	Mean sum squares of forecast errors
__rmspe	Root mean absolute percent of forecast errors.
__u1	Theil inequality coefficient (0<=__u1<=1)
__uc	Covariance proportion of mean sum squares of errors.
__ue	Disturbance proportion of mean sum squares of errors.
__um	Bias proportion of mean sum squares of errors.
__ur	Regression proportion of mean sum squares of errors.
__us	Variance proportion of mean sum squares of errors.
__vf	Variance of predicted or forecast series.

Note: FORECAST is not available for nonlinear models.

Miscellaneous

A few GAUSS built-in procedures have been modified, and they can be called throughout the program using GPE package.

Procedure	Description
gradp2	Computes the gradient vector or matrix (Jacobian) of a scalar- or vector-valued function that has been defined in a procedure. The GPE procedure gradp2 works the same as GAUSS built-in procedure gradp except that gradp2 takes three input entries and has the format: g = gradp2(&f,x,b). &f is a pointer to a vector-valued function f(x,b) defined as a procedure, x is a data matrix, and b is a vector of points at which to compute gradient. Both x and b are used to define the function f. The output of gradp2 is the same as that of gradp, which is a vector or matrix of first derivatives of function f evaluated at b. See GAUSS Language References or on-line help for more details about GAUSS built-in

procedure gradp.

`hessp2` Computes the matrix of second partial derivatives (Hessian matrix) of a scalar-valued function that has been defined in a procedure. The GPE procedure hessp2 works the same as GAUSS built-in procedure hessp except that hessp2 takes three input entries and has the format: h = hessp2(&f,x,b). &f is a pointer to a vector-valued function f(x,b) defined as a procedure, x is a data matrix, and b is a vector of points at which to compute Hessian. Both x and b are used to define the function f.
The output of hessp2 is the same as that of hessp, which is the matrix of second derivatives of function f evaluated at b. See GAUSS Language References or on-line help for more details about GAUSS built-in procedure hessp.