WinSolve Version 4: Log File Language

Richard G. Pierse

November 2007

This is a formal description of the command language used by WinSolve log files. The first line of a log file must start with the characters 'WinSolve log file' (without the quotes). Commands and keywords are case insensitive and keywords can appear in any order. Each command should be on a single line unless otherwise stated. Lines starting with the 'at' character @ are treated as comments and ignored. The final command should be the EndLog command or the Exit command if the log file is to be run in batch mode only.

Conventions used in this description:

bold	typeface indicates a command or keyword that should be typed as shown
italic	typeface indicates a parameter or value to be supplied by the user
[ ]	denotes an optional keyword or parameter which may be omitted
|	denotes a set of (mutually exclusive) alternative keywords
. .	(ellipsis) denotes input is a series of values as indicated
//	indicates that the rest of the command should continue on a new line
date1– date2	denotes a pair of valid WinSolve dates separated by a minus sign
pathname	denotes a pathname to a file. If a full path is not included the current directory will be assumed
vname	denotes the name of a variable in the current model

List of Commands

Model Object Commands

ModelOpen *pathname* [**desc** *str*]

Open a model object from file *pathname* with optional model description *str*.

ModelSave *pathname*

Save the current model object to file *pathname.*

ModelClose

Closes the current model object.

Data Set Commands

DataNew **freq** *f* **per** *date1–date2* [**init** *z*]

Create a new data set over the period *date1–date2* with frequency *f*.
Possible values for *f* are 1(annual or undated), 4 (quarterly) or 12 (monthly).
init Initialise all observations to value *z.*

DataOpen *pathname* [**merge1** | **merge2**]

Read a new data set from file *pathname*. (Default is to replace current data set)
merge1 merge data set with current data set (new values taking precedence)
merge2 merge data set with current data set (old values taking precedence)

DataEdit *vname* **per** *date1–date2* // *val1* **..** *valn*

Replace observations on variable *vname* over the period *date1–date2* with the *n* values *val1* to *valn*

DataProj *vname* **per** *date1–date2* [**cnst** *a*] [**trnd** *b*] [**ldv** *c*] [**rndm** *d*]
[**repl** | **add** | **mult**]

Project observations on variable *vname* over the period *date1–date2* using a simple model of the form: *x = a + b*t +c*x(–1) + d*u where u~N(0,1).*

cnst include a constant with coefficient *a*
trnd include a time trend with coefficient *b* (*t=1* in period *date1*)
ldv include a lagged dependent variable with coefficient *c*
rndm include a stochastic error with coefficient *d*
repl replace current values with projected process (default)
add add projected process to current values
mult multiply current values by projected process

DataIAR *vname* **est** *date1–date2* **proj** *date3–date4* [**order** *p*] [**dif** *d*]
[**levels | logs**] [**cnst**] [**trnd**] [**seas**] [**rndm**]

Estimate an (integrated) autoregressive ARIMA(p,d,0) process for variable *vname* over the period *date1–date2* and use this process to project the variable over the period *date3–date4*.

order lag order of the estimated AR process (default 1)
dif integration order of the estimated process (default 0)
levels estimate the process in the levels of the variables (default)
logs estimate the process in the logarithms of the variables
cnst include a constant in the regression
trnd include a time trend in the regression
seas include deterministic seasonals in the regression
rndm add a stochastic error to the projected process

DataReset *vname* | **all** *rname* | **zero**

Reset model data on variable *vname* (or *all* variables)
zero set data to zero
rname set data to solution values from run *rname*

DataSave *pathname*

Save the current data set to file *pathname*. The file suffix indicates the data format.

Model Adjustment Commands

AdjOpen *pathname* [**merge1** | **merge2**]

Read a new set of model adjustments from file *pathname*. (Default is to replace current adjustments)

merge1 merge with current adjustments (new values taking precedence)
merge2 merge with current adjustments (old values taking precedence)

AdjEdit *vname* **per** *date1–date2* // *val1* **..** *valn*

Replace model adjustments on variable *vname* over the period *date1–date2* with the *n* values *val1* to *valn*

AdjProj *vname* **per** *date1–date2* [**cnst** *a*] [**trnd** *b*] [**ldv** *c*] [**rndm** *d*] [**repl** | **add** | **mult**]

Project model adjustments on variable *vname* over the period *date1–date2* using a simple model of the form: $x = a + b*t + c*x(-1) + d*u$ *where* $u \sim N(0,1)$.

cnst include a constant with coefficient *a*
trnd include a time trend with coefficient *b* (*t=1* in period *date1*)
ldv include a lagged dependent variable with coefficient *c*
rndm include a stochastic error with coefficient *d*
repl replace current values with projected process (default)
add add projected process to current values
mult multiply current values by projected process

AdjIAR *vname* **est** *date1–date2* **proj** *date3–date4* [**order** *p*] [**dif** *d*] [**levels | logs**] [**cnst**] [**trnd**] [**seas**] [**rndm**]

Estimate an (integrated) autoregressive ARIMA(p,d,0) process for the adjustments on variable *vname* over the period *date1–date2* and use this process to project the adjustments over the period *date3–date4*.

order lag order of the estimated AR process (default 1)
dif integration order of the estimated process (default 0)
levels estimate the process in the levels of the variables (default)
logs estimate the process in the logarithms of the variables
cnst include a constant in the regression
trnd include a time trend in the regression
seas include deterministic seasonals in the regression
rndm add a stochastic error to the projected process

AdjReset *vname* | **all** *rname* | **zero**

Reset model adjustments on variable *vname* (or *all* variables)
zero set adjustments to zero
rname set adjustments to adjustments from run *rname*

AdjDefType int | add | mult

Reset the default adjustment type
int default adjustment type is intelligent
add default adjustment type is additive
mult default adjustment type is multiplicative

AdjType *vname* **def | add | mult**

Reset the adjustment type on variable *vname*
def adjustment type is default type
add adjustment type is additive
mult default adjustment type is multiplicative

AdjSave *pathname*

Save the current set of model adjustments to file *pathname*. The file suffix indicates the data format.

Model Assumptions Commands

Fix [*vname* | **all**] **per** *date1–date2* [**inst** *iname* [**mult** z]] [**unset**]

Fix (exogenenise) variable *vname* (or *all* variables) over period *date1–date2*
inst Type 2 fix using variable *iname* as instrument with multiplier *z* (default 1.0)
unset *Do not* reset adjustments on variable to zero (default is to reset adjustments)

Unfix [*vname* | **all**] **per** *date1–date2*

Unfix (re-endogenenise) variable *vname* (or *all* variables) over period *date1–date2*

Alt *vname n*

Select alternative equation number *n* for variable *vname*

Term *vname* [**const** | **exog** | **grow** | **agrow** | **lgrow** | **adif** | *n*]

Select terminal condition for variable *vname*. The terminal condition is one of:

const	constant level
exog	exogenous value
grow	constant period-on-period growth rate
agrow	constant year-on-year growth rate
lgrow	constant long run growth rate
adif	constant year-on-year difference
n	user-specified alternative terminal condition number *n*

OrderOpen *pathname*

Reorder model equations using equation order defined in file *pathname*

OrderSave *pathname*

Save current equation ordering to file *pathname*

Rand [**seed** *seedval* | **null**] [**disable** | **enable**]

Set random number generator options
seed reseed random number generator to *seedval* or **null** (seed generated automatically from date/time stamp)
disable switch off generation of random numbers
enable re-enable generation of random numbers

Model Solution Commands

SolveRun *rname* **per** *date1–date2* [**mode**] [**noragged**] [**norat**] [**monitor**] [**last**]

Starts a new model solution with run name *rname* over the period *date1–date2*.

mode is the mode of model solution and can be one of:
dynsys dynamic system solution (the default)

static	static system solution
dynsingle	dynamic single equation solution
single	single equation solution
steady	solve for steady state
base	create a simulation base

noragged	let equations take precedence over data (disable ragged edge)
norat	solve the model without rational expectations
monitor	monitor solution by graphing variables in real-time
last	start solution from final solution point of last run

If **monitor** is selected then this command must be immediately followed by a **Monitor** command.

Monitor *vname1 type1* **. .** *vnamek typek***;**

Monitor in real-time the *k* variables *vname1* to *vnamek* where *type* can be one of:

sol	Simulation (solution) values
dif	Differences of solution from base
pct	Percentage differences of solution from base

The **Monitor** command can only appear immediately following a **SolveRun** or **ControlRun** command that has selected the **monitor** option.

SolveOpt [**alg** *alg*] [**maxerr** *n*] [**errval** *f*] [**itmax** *n*] [**itp** *n*] [**abs** *f*] [**pct** *f*] [**alpha** *f*] [**beta** *f*]

Set options relating to the model solution algorithm

alg is the solution algorithm. *alg* can be one of:

gausseidel	Gauss-Seidel (the default)
fgs	Fast Gauss-Seidel
jacobi	Jacobi algorithm
newton	Newton's method (with analytic derivatives)
sparse	Newton's method using sparse matrix techniques

maxerr	maximum number of errors before solution is abandoned (default 0)
errval	value to substitute for illegal argument in case of error (default 1.0)
itmax	maximum number of iterations (default 100)
itp	iteration to commence printing unconverged variables (default 100)
abs	absolute convergence criterion (default 0.025)
pct	percentage convergence criterion (default 0.025)
alpha	relaxation parameter (default 1.0)
beta	relaxation parameter for FGS algorithm (default 1.0)

ExpOpt [**alg** *alg*] [**maxerr** *n*] [**errval** *f*] [**itmax** *n*] [**itp** *n*] [**abs** *f*] [**pct** *f*] [**alpha** *f*] [**jit**]

Set options relating to the expectations algorithm (for models including leads)

alg is the solution algorithm and can be one of:

fairtaylor	Fair-Taylor (the default)
stacked	Stacked Newton algorithm
diy	User-specified customised algorithm

maxerr maximum number of errors before solution is abandoned (default 0)
errval value to substitute for illegal argument in case of error (default 1.0)
itmax maximum number of iterations (default 100)
itp iteration to commence printing unconverged variables (default 100)
abs absolute convergence criterion (default 0.025)
pct percentage convergence criterion (default 0.025)
alpha expectations loop relaxation parameter (default 1.0)
jit Selects 'just in time' updating of the model terminal conditions

OptVar *vname* [**abs** *tol1*] [**pct** *tol2*]

Set (relative) convergence criteria for variable *vname*
abs set absolute convergence criterion to *tol1* (default 1.0)
pct set percentage convergence criterion to *tol2* (default 1.0)

MakeBase

Make the last solution into a simulation base

SelectDIY *pathname*

Select a file of diy commands from the file *pathname*.

StochRun *rname* **nrep** *nrep* **per** *date1–date2* [**seed** *seedval*] [**gen** *date3–date4*] [**app** *date5–date6*] [method] [**df** *dfval*] [**anti**] [**quant** *nq*] [**filename** *fn1*] [**savefile** *fn2*]

Start a stochastic simulation with run name *rname* and *nreps* replications over the solution period *date1–date2*. Shocks are generated from an initial seed *seedval* using residuals over the period *date3–date4* and applied over the period *date5–date6*.

method is the method used to generate shocks and can be one of:

cholesky Cholesky method
mccarthy McCarthy method
bootstrap bootstrap method
user user-specified variance matrix
file shocks read from user-specified file

If **user** is selected, then this command must be immediately followed by a **UserVar** command.

df apply a small-sample degrees of freedom correction *dfval*
anti use antithetic shocks
quant compute *nq* simulation quantiles (default 0).
filename reads shocks from file *fn1* (must be specified when method is **file**)
savefile saves replication results to file *fn2*.

(Note that the file extension specified in the **filename** or **savefile** options must be correspond to a file type that is organised by observation e.g. .csv, .xls, .wk1, .dat.)

UserVar *rname1 cname1 val1* **..** *rnamek cnamek valk* **;**

Specify nonzero elements of variance-covariance matrix. This command should follow *immediately* after the **StochRun** command when the **user** method has been selected. There are *k* sets of arguments of the form { *rnamei cnamei vali* } where *rnamei* is the name of the row variable, *cnamei* is the name of the column variable and *vali* is the corresponding element of the variance covariance matrix. Note that WinSolve automatically imposes symmetry so that the user need only specify the off-diagonal elements above (or below) the diagonal. The command is terminated with a semi-colon.

Optimal Control Commands

ControlNew [**nplay** *n*] [**user | lquad | nls**] [**maxsv** *n*] [**maxcv** *n*] [**per** *date1–date2*] [**disc** *n*] [**penalty**]

Create a new optimal control game to be solved over the period *date1–date2*

nplay number of game players (default 1)
user user-defined loss function (default)
lquad linear-quadratic loss function
nls non-linear quadratic (optimal regulator) loss function
maxsv maximum number of target variables of any player (default 10)
maxcv maximum number of control variables of any player (default 10)
disc time discount rate (default 1.0)
penalty penalise changes in control variables

ControlOpen *pathname*

Open a saved control game from file *pathname*.

ControlClose

Closes an open control game.

LQMatrix **Q | R | S** [**player** *i*] // *rname1 cname1 val1* **. .** *rnamen cnamen valn* ;

Define non-zero elements of coefficient matrices in the loss function for player *i* (default 1) in linear-quadratic control game

Q coefficients on target variables (specify upper or lower triangle only)
R coefficients on control variables (specify upper or lower triangle only)
S cross-coefficients between targets and controls
rnamei name of *ith* row variable of non-zero coefficient
cnamei name of *ith* column variable of non-zero coefficient
vali value of non-zero *ith* element

Targets [**player** *i*] // *vname1* **. .** *vnamen* ;

Define names of target variables for player *i* (default 1) in the control game.

Controls [**player** *i*] // *vname1* **. .** *vnamen* ;

Define names of control variables for player *i* (default 1) in the control game.

TargetEdit *vname* **per** *date1–date2* [**player** *i*] // *val1* **. .** *valn* ;

Set target values for variable *vname* for player *i* (default 1) over the period *date1–date2* to the *n* values *val1* to *valn*

TargetProj *vname* **per** *date1–date2* [**cnst** *a*] [**trnd** *b*] [**ldv** *c*] [**rndm** *d*]
[**repl** | **add** | **mult**] [**player** *i*]

Project target values on variable *vname* for player *i* (default 1) over the period *date1–date2* using a simple model of the form: *x = a + b*t +c*x(–1) + d*u where u~N(0,1).*

cnst include a constant with coefficient *a*
trnd include a time trend with coefficient *b* (*t=1* in period *date1*)
ldv include a lagged dependent variable with coefficient *c*
rndm include a stochastic error with coefficient *d*
repl replace current values with projected process (default)
add add projected process to current values
mult multiply current values by projected process

TargetIAR *vname* **est** *date1–date2* **proj** *date3–date4* [**order** *p*] [**dif** *d*]
[**levels | logs**] [**cnst**] [**trnd**] [**seas**] [**rndm**] [**player** *i*]

Estimate an (integrated) autoregressive ARIMA(p,d,0) process for target variable *vname* for player *i* (default 1) over the period *date1–date2* and use this process to project the variable over the period *date3–date4*.

order lag order of the estimated AR process (default 1)
dif integration order of the estimated process (default 0)
levels estimate the process in the levels of the variables (default)
logs estimate the process in the logarithms of the variables
cnst include a constant in the regression
trnd include a time trend in the regression
seas include deterministic seasonals in the regression
rndm add a stochastic error to the projected process

WeightEdit *vname* **per** *date1–date2* [**player** *i*] // *val1* **..** *valn*

Set welfare function weights on target variable *vname* for player *i* (default 1) over the period *date1–date2* to the *n* values *val1* to *valn*

WeightProj *vname* **per** *date1–date2* [**cnst** *a*] [**trnd** *b*] [**ldv** *c*] [**rndm** *d*]
[**repl** | **add** | **mult**] [**player** *i*]

Project welfare function weights on target variable *vname* for player *i* (default 1) over the period *date1–date2* using a simple model of the form: *x = a + b*t +c*x(–1) + d*u where u~N(0,1).*

cnst include a constant with coefficient *a*
trnd include a time trend with coefficient *b* (*t=1* in period *date1*)
ldv include a lagged dependent variable with coefficient *c*
rndm include a stochastic error with coefficient *d*
repl replace current values with projected process (default)
add add projected process to current values
mult multiply current values by projected process

WeightIAR *vname* **est** *date1–date2* **proj** *date3–date4* [**order** *p*] [**dif** *d*]
[**levels | logs**] [**cnst**] [**trnd**] [**seas**] [**rndm**] [**player** *i*]

Estimate an (integrated) autoregressive ARIMA(p,d,0) process for welfare function weights on target variable *vname* for player *i* (default 1) over the period *date1–date2* and use this process to project the variable over the period *date3–date4*.

order lag order of the estimated AR process (default 1)
dif integration order of the estimated process (default 0)
levels estimate the process in the levels of the variables (default)
logs estimate the process in the logarithms of the variables
cnst include a constant in the regression
trnd include a time trend in the regression
seas include deterministic seasonals in the regression
rndm add a stochastic error to the projected process

ControlRun *rname* **per** *date1–date2* [**mode**] [**noragged**] [**norat**] [**icred** *n*] [**monitor**] [**last**]

Run an optimal control game with run name *rname* over the period *date1–date2*. Parameters are the same as for the SolveRun command except for **icred**

icred = 0 optimal control (the default):
= 1 time-consistent control (policy makers Stackelberg leaders)
= 2 time-consistent control (policy makers in Nash game with economy)

ControlOpt [**alg** *alg*] [**disc** *n*] [**ntp** *n*] [**itmaxg** *n*] [**itmaxp** *n*] [**gtol** *f*] [**ptol** *f*] [**ltol** *f*]

Set options relating to optimal control solution algorithm

alg is the optimisation algorithm for the player loss function and can be one of:
newton BFGS quasi-Newton algorithm (default)
gaussn Gauss-Newton algorithm with numerical derivatives
gaussa Gauss-Newton algorithm with analytic derivatives
powell Powell conjugate directions algorithm

disc time discount rate (default 1.0)
ntp time interval between updates of controls (default 1 i.e. every period)
itmaxg maximum number of game iterations (default 100)
itmaxp maximum number of iterations in player optimisation (default 500)
gtol tolerance in game iteration loop (default 0.025)
ptol tolerance in player iteration loop (default 0.025)
ltol tolerance in line search in player optimisation (default 0.025)

ControlSave *pathname*

Saves a control game to file *pathname*.

View Results Commands

ViewNew // > *title* // **per** *date1–date2* [initial] [**file** *fn*] //
name1 typefreqtran1 rname1 bname1 **. .** *namek typefreqtrank rnamek bnamek* **;**

Create a new table object over the period *date1–date2* with a description given by string *title* (on a new line and preceded by a > character).
initial defines the initial view of the object and can be one of
graph view as a graph

table view as a table
stats view summary statistics
spread create object as spreadsheet

file save the table object to file *fn*

The variables to be included in the table are specified as quadruplets {*name typefreqtran rname bname*} where *name* is the variable name, *rname* is the run name (or **latest** to signify the most recent run), *bname* is the base run name (or **base** to signify the data or simulation base) and *typefreqtran* is a string specifying the type, frequency and transformation of the variable.

The string *typefreqtran* is made up by concatenating one of the following type codes:

sol	Simulation (solution) values
bas	Base (data) values
adj	Adjustments
res	Implicit residual values
dif	Differences of solution from base run *bname*
cal	Calculated values (solution values minus implicit residuals)
pct	Percentage differences of solution from base run *bname*
ctg	Cost to go (linear-quadratic control run only)
ctr	Cost of reneging (linear-quadratic control run only)
sdv	Standard deviations of simulation (stochastic simulation run only)
+5%	Positive 5% standard error band (stochastic simulation run only)
–5%	Negative 5% standard error band (stochastic simulation run only)
skw	Skews of simulation (stochastic simulation run only)
kur	Kurtosis of simulation (stochastic simulation run only)
mod	Mode of simulation (stochastic simulation run only)
fan	Modal fanchart (stochastic simulation run only)
qua	Quantile fanchart (stochastic simulation run only)
q*i*	*i*% quantile of simulation (e.g. q50 for median, q25 for 1st quartile)

with (optionally) one of the six frequency codes:

sum	Calendar year sum
ave	Calendar year average
end	Calendar year end value
fsum	Financial year sum
fave	Financial year average
fend	Financial year end value

and (optionally) one of the seven transformation codes:

dif	First difference of variable
%dif	Percentage first difference of variable
adif	Annual difference of variable
%adif	Percentage annual difference of variable
hpd	De-trended variable (de-trending done with Hodrick-Prescott filter)
hpu	Undetrended variable (inverting the Hodrick-Prescott filter)
hpt	Trend of variable (trend from Hodrick-Prescott filter).

The sequence of quadruplets may run over more than one line and the last quadruplet should be followed by a semicolon, to denote the end of the command.

ViewOpen *pathname*

Open a predefined table from table definition file *pathname* and display it.

ViewSave *pathname* [**append**] // > *title* // **per** *date1–date2* [initial] [**file** *fn*] //

name1 typefreq1 rname1 **..** *namek typefreqk rnamek* **;**

Save a table definition to file *pathname*
append append table to end of file (default is to overwrite file)

Other parameters are the same as for the **ViewNew** command.

RunLoad *pathname*

Loads a run from the file *pathname*. The name of the run will be the same as the file name (omitting the path and suffix)

RunSort [**del** *rname*] [**ren** *rname* *newrname***]**

Sort results runs. Allows runs to be deleted or renamed.
del Delete run *rname*
ren Rename run *rname* as run *newrname*

RunComp *rname1 rtype1 rname2 rtype2* **per** *date1–date2* **[abs** *abs***] [pct** *pct***]**

Compares values from runs *rname1* and *rname2* over the period *date1–date2* using absolute tolerance *abs* and percentage tolerance *pct* (defaults solution tolerances).
The file types *rtype1* and *rtype2* can be one of:
sol Simulation (solution) values
adj Model adjustments
sdv Standard deviations of simulation (stochastic simulation run only)

RunSave *pathname* **per** *date1–date2* **run** *rname* [**sol | dif | pct | sdv**]

Save all results from run *rname* to file *pathname* over the period *date1–date2*.
sol Save solution values (default)
adj Save model adjustments
dif Save differences of solution from base
pct Save percentage differences of solution from base
sdv Save standard deviations of simulation (stochastic simulation run only)

Print

Print the currently selected window.

EndLog

End of log file.

Exit

Terminate log file and exit from WinSolve. For running in batch mode only.

#LOGCOM
$ Log File Language
#LOGDEF
k Log File Definition
#LOGMODEL
k Model Object log commands
#LOGMODELOPEN
k ModelOpen log command
#LOGMODELSAVE
k ModelSave log command
#LOGMODELCLOSE
k ModelClose log command
#LOGDATA
k Data Set log commands
#LOGDATANEW
k DataNew log command
#LOGDATAOPEN
k DataOpen log command
#LOGDATAEDIT
k DataEdit log command
#LOGDATAPROJ
k DataProj log command
#LOGDATAIAR
k DataIAR log command
#LOGDATARESET
k AdjReset log command
#LOGDATASAVE
k DataSave log command
#LOGADJ
k Model Adjustment log commands
#LOGADJOPEN
k AdjOpen log command
#LOGADJEDIT
k AdjEdit log command
#LOGADJPROJ
k AdjProj log command
#LOGADJIAR
k AdjIAR log command
#LOGADJRESET
k AdjReset log command
#LOGADJDEFTYPE
k AdjDefType log command
#LOGADJTYPE
k AdjType log command
#LOGADJSAVE
k AdjSave log command
#LOGASS
k Model Assumptions log commands
#LOGFIX
k Fix log command
#LOGUNFIX
k UnFix log command
#LOGALT
k Alt log command
#LOGTERM
k Term log command
#LOGORDEROPEN
k OrderOpen log command

#LOGORDERSAVE
k OrderSave log command
#LOGRAND
k Rand log command
#LOGSOL
k Model Solution log commands
#LOGSOLVERUN
k SolveRun log command
#LOGMONITOR
k Monitor log command
#LOGSOLVEOPT
k SolveOpt log command
#LOGEXPOPT
k ExpOpt log command
#LOGOPTVAR
k OptVar log command
#LOGMAKEBASE
k MakeBase log command
#LOGSELECTDIY
k SelectDIY log command
#LOGSTOCHRUN
k StochRun log command
#LOGDIAGVAR
k DiagVar log command
#LOGCON
k Optimal Control log command
#LOGCONTROLNEW
k ControlNew log command
#LOGCONTROLOPEN
k ControlOpen log command
#LOGCONTROLCLOSE
k ControlClose log command
#LOGLQMATRIX
k LQMatrix log command
#LOGTARGETS
k Targets log command
#LOGCONTROLS
k Controls log command
#LOGTARGETEDIT
k TargetEdit log command
#LOGTARGETPROJ
k TargetProj log command
#LOGTARGETIAR
k TargetIAR log command
#LOGWEIGHTEDIT
k WeightEdit log command
#LOGWEIGHTPROJ
k WeightProj log command
#LOGWEIGHTIAR
k WeightIAR log command
#LOGCONTROLRUN
k ControlRun log command
#LOGCONTROLOPT
k ControlOpt log command
#LOGCONTROLSAVE
k ControlSave log command
#LOGVIEW
k View Results log commands
#LOGVIEWNEW

[k] ViewNew log command
[#]LOGVIEWOPEN
[k] ViewOpen log command
[#]LOGVIEWSAVE
[k] ViewSave log command
[#]LOGRUNLOAD
[k] RunLoad log command
[#]LOGRUNSORT
[k] RunSort log command
[#]LOGRUNCOMP
[k] RunComp log command
[#]LOGRUNSAVE
[k] RunSave log command
[#]LOGPRINT
[k] Print log command
[#]LOGENDLOG
[k] EndLog log command
[#]LOGEXIT
[k] Exit log command