Methodological Appendices

MA1: Unit Roots in Pooled Time-Series-Cross-Section Data

In the single time-series context the potential problem of *unit roots* in one's data is fairly well understood; the potential solutions to the problem, as in the usual manner of things, are less so. One of the most active areas of current methodological research in the social sciences, especially in econometrics, is precisely this topic. To avoid excessive formality, and to get to the point of interest here, let me first merely summarize the more or less generally accepted and understood wisdom.[1]

The basic issues of unit roots are two: one statistical and one inferential. The statistical issue is that, in a time series model in which the serial correlation(s) of the stochastic process(es) is (sum to) one, the usual limiting distribution theory on which we base our estimates of standard errors and our hypotheses tests will not apply. The inferential issue is that, in regressions involving variables on the left-hand and the right-hand side with unit roots, we will tend to find relationships (strong partial correlations) in limited samples which do not in fact describe *systematic* features of the data. Let me explain the statistical issue first, beginning with a quick simplified version the standard discussion.

$$(1)\quad y_t = \rho\, y_{t-1} + \varepsilon_t$$

Suppose the variable of interest, y, evolves over time according to the process described in equation 1, where ε_t is i.i.d. white noise. Then by recursively substituting for y_{t-i} we can rewrite the process as:

$$(2)\quad y_t = \sum_{i=0}^{\infty} \rho^i\, \varepsilon_{t-i}$$

Since this is so,[2] we can write the variance of *y* as

[1] I'd like to thank Jim Alt, Neal Beck, Simon Jackman, Jonathan Katz, Gary King, Brad Palmquist, Doug Rivers, Renée Smith, and Duane Swank for various conversations on this and related topics which they will likely not recall and almost certainly will not recognize as having contributed to any of this: nonetheless, nearly every word was indispensable to whatever understanding I may now have of these issues. (Thus, more than the usual *caveat* about all the blame for remaining idiocies belonging with me applies.)

[2] Assume for the moment *y* goes back infinity periods just to keep things illustratively clear.

$$(3) \quad V(y) = V(\sum_{i=0}^{\infty} \rho^i \ \varepsilon_{t-i}) = \sum_{i=0}^{\infty} (\rho^i)^2 \ V(\varepsilon_{t-i}) = (\frac{1}{1-\rho})^2 \ \sigma_\varepsilon^2$$

where σ^2 is the variance of the individual ε's and the last equation holds for all $|\rho|<1$. If, instead, $\rho>1$, then the series y is explosive; any arbitrarily small shock causes y to head off into infinity. If $\rho<-1$, the series is also explosive; any arbitrary small shock sends y into ever larger oscillations off into infinity. Alternatively expressed, for either case a shock 5000 (or any number of) years ago has more impact on today's observation than one yesterday. Obviously, neither makes any sense as a description of variables of interest to social scientists.[3]

When $\rho=1$ (or -1), the variance of y is undefined. This is the crux of the "unit-root problem." Because the variance is undefined in the limit or, more generally, because it depends on time, our standard t-, F- and other testing distributions will not appear even asymptotically when we need them if there are unit roots involved.

There may be statistical and inferential problems with estimating the usual sorts of linear time-series models with lagged values of the dependent variable on the left-hand side (or and AR-type model. The statistical issues involve the fact that, if the coefficient(s) is (sum to) 1, then the effect of some stochastic shock today never dissipates as it would in a series where the coefficient is less than 1. It would follow that the variance of (the stochastic component of) TT would not be constant (as assumed in the simple regression model), but would instead be an increasing function of time. Thus, the usual (limiting) distribution theories on which we base hypotheses tests would not be valid. The inferential issue is that, in limited samples, an I(1) series[4] will tend to drift in one direction or another. However, if one can find a set of variables that *cointegrate* with the I(1) dependent variable, y, that is an I(1) set of variables, X, that move in equilibrium with the dependent variable such that $y\text{-}E(y|X)$ is stationary, inference can proceed as usual.

In model specifications of the sort used in the text, some linear combination of the right-hand-side variables may serve this purpose. If they do, then the coefficient on the lagged level of the dependent variable will be *appreciably* below zero. One says *appreciably* rather than *significantly* here to emphasize that, if the dependent variable does possess a unit root, and if right-hand-side variables do not offer sufficient

[3] We should probably take any such findings as indicative of mis-specification; I'll return to this point.

[4] A series with a single unit root is said to be "integrated of order one" or I(1). The *unit* of unit roots comes from the coefficient on one on the lagged dependent variable (and from the eigenvalues of the characteristic equation). The reference to *integrated* comes from the fact that the current value of the variable can be expressed as the sum (integration) of all previous stochastic shocks. The *order one* comes from the number of times the variable has to be differenced (the opposite of summed or integrated) to make it *stationary* (lack a unit root).

cointegration, then *y-E(y|X)* remains non-stationary, which implies that the usual *t-statistics*, of coefficients divided by standard errors, are not in fact *t* distributed. Like the statistics from ADF tests, they are distributed Dickey-Fuller, which distribution changes with the number and nature of right-hand-side terms and the number of observations. That distribution being therefore indeterminate in the cases in the book, footnotes in the text suggest that a "*t*-statistic" well over 3 or 4 will probably suffice. Those numbers reflect the author's crude guess based on experience with standard ADF tests the distribution of which is known, which typically suggest coefficients-divided-by-standard-errors in excess of at least 3 and sometimes as much as 4 or a little more, suffice.

MA2: Interpreting Interaction Coefficients and Testing Hypotheses About Them

Please See:

Franzese, R., C. Kam, A. Jamal. 1999. "Modeling Interactive Hypotheses and Interpreting Statistical Evidence Regarding Them," currently available at: *http://www-personal.umich.edu/~franzese/interactions.pdf*.

and

Franzese, R., C. Kam. 2002. "Modeling Interactive Hypotheses in Regression Analysis: A Brief Refresher and Some Practical Advice," currently available at: *http://www-personal.umich.edu/~franzese/InteractionsPaper.pdf*.

MA3: Some Comments on Vector Auto-Regression (VAR) in Pooled Time-Series-Cross-Section (TSCS) Data

For VARs estimated from TSCS data as if they were simple time-series data to have optimal properties requires stringent assumptions. However, these assumptions are no more stringent than those required for OLS to have optimal properties under such conditions. To see this, note that VARs assume that, controlling for all the lags as they do, the residuals are *i.i.d.* within equation, although they may have covariance across equations (thus the need for ordering and Cholesky decomposition). They also assume that the coefficients on the lags correctly describe (in summary reduced form) the responses of each variable to past movements in itself and the other variables. Thus, provided that when we estimate VARs from TSCS data we ensure that "lags" do not operate across cross-sections (*i.e.*, we ensure that X_{t-4}, say, does not cross from, say, t=Japan 1952 to t-1=Japan 1951 to t-2=Japan 1950, to t-3=*US* 1999 to t-4=US 1998--and inserting sufficient missing data between cross-sections can assure this), this is no more nor less than what we one assumes in analyzing TSCS data as time-series data. Recall also the key point from Freeman *et al.* 1996 that VARs are weak (prone to fail to reject false nulls), but do not over-reject in most cases. FM-VAR is weaker still, but may be less prone to over-reject in the presence of unit roots and absence of cointegration where regular VAR over rejects by 2-3 times depending on sample size. We have good reason to trust that our examples do not have unit roots without cointegration as I have noted in the text.

MA4: A Gauss Procedure to Estimate Panel-Corrected-Standard-Errors with Non-rectangular and/or Missing Data [1]

Econometric analysis of time-series-cross-section (TSCS) data often present the researcher with residuals which exhibit contemporaneous correlation (i.e., residuals from different cross-sections (CS's) in the same time period (TP) are correlated) in addition to the more usual time-series property of serial (or auto-) correlation and the typical cross-section property of heteroskedasticity. This non-sphericity in the error-covariance matrix implies that estimated coefficients are inefficient and that standard errors are inconsistent. The usual solution, prior to Beck and Katz (1995, 1997) was to estimate Feasible Generalized Least Squares in up to three steps, estimating a serial-correlation parameter for each cross-section, a error-variance parameter for each cross-section, and a number of contemporary correlation parameters. Asymptotically, this solution is unbiased, consistent, and efficient in both its coefficient and standard-error estimates.

Beck and Katz (1995, 1997) show, however, that in more realistically sized data-sets the procedure tends to seriously under-estimate standard errors. In essence, the problem is that there are a large number of parameters in the error-covariance matrix to be estimated and relatively few observations are being leveraged to estimate them. While this is perfectly fine in that mythical land of *asymptopia*,[2] in practical samples it has a tendency to overstate one's certainty about coefficient estimates. That is, it tends to over-fit the data, or more precisely to over-fit patterns in the residuals, relative to what is truly optimal in a random sample of samples.

They suggest, therefore, that one deal with the serial-correlation property separately, transforming the data and/or including lag(s) prior to worrying about contemporaneous correlation.[3] Panel heteroskedasticity, perhaps should be ignored; the implications of their findings regarding such non-constant variance by CS depend on the actual sample size and the researcher's distaste for inefficiency relative to standard-error inaccuracy. In any event, it may be treated in the usual WLS way or ignored, and, while they suggest the latter is usually safer for smaller data sets, the

[1] This text of this appendix is only a minor rewriting of my notice in *The Political Methodologist* (1996). Gratitude is due to Neal Beck for providing the Beck-Katz procedures, for other suggestions, and for general helpfulness. This piece has also benefitted from the helpful criticism of James Alt, Gary King, Kenneth Scheve, and Curtis Signorino. Special thanks to Curt for his assistance in optimizing the procedure for more minimal memory and time use and to Kenneth Scheve for finding an error in an earlier version of the procedure.

[2] Actually it is fine only in one province of asymptopia: the one where the number of TPs goes to infinity.

[3] Actually, they make a stronger argument, based on the same reasoning and findings as their argument regarding contemporaneous correlation as described in the previous paragraph, that estimating a single serial-correlation parameter rather than N different parameters is probably optimal in small (*i.e.*, real) samples.

issue is separate from what should be done about contemporaneous correlation. These findings are treated in more detail in the *Political Analysis* (forthcoming). Finally, and most importantly for the present purposes, they show (through Monte Carlo experiments) that the efficiency gains from the contemporaneous-correlation correction are typically small to non-existent while the standard-error "bias" is relatively large. Accordingly, they suggest estimating standard errors according to a robust procedure which incorporates potential contemporaneous-correlation information into the coefficient variance-covariance matrix without adjusting the coefficient estimate. They show that this procedure--controlling for serial correlation in some (simple) way prior to final estimation and relying on robust standard errors, i.e. Panel-Corrected Standard-Errors or PCSE's--has admirable small sample properties relative to alternatives.

Beck and Katz have also made available computer programs for calculating these PCSE's in RATS or GAUSS. However, both of their procedures constrain the data to be analyzed to be rectangular (*i.e.*, all CS's begin and end in the same year) and to have no missing data. This is a restriction in their programs, not in their econometric analysis. In other words, in principle, there is no reason PCSE's cannot be estimated in non-rectangular samples and/or samples with missing data. This is fortunate since almost every pooled data set I have encountered has at least one or the other complication. For example, in Chapter III, I had a data set with 20 CS's and up to 35 TP's, but sometimes as few as 20 and with a couple very unfortunately placed[4] missing data points, for a total of 550+/- usable observations, but if I had to rectangularize it somehow and remove breaks in the sample within a CS, it would have had at most 300 observations! An extreme example to be sure, but it is not rare to have to sacrifice 15% or more of one's data to rectangularize it and remove missing-data breaks. Of course, we would like to avoid this if possible. Accordingly, I have written a GAUSS procedure, *pcse.g*, which will accept such data and produce PCSE estimates along with the usual regression output. The procedure also adds a number of options which I think researchers commonly dealing with TSCS data will find useful.

Beck and Katz's PCSE estimate of coefficient standard errors is defined by

$$(X'X)^{-1}X'\left(\frac{E'E}{T}\otimes I_t\right)X(X'X)^{-1}$$

where E is the TxN (number of TP's by number of CS's) matrix of OLS residuals, $\otimes$ is the Kroenecker product operator, and I_t is an identity matrix of size T. Notice that E'E/T is just an estimate of the error covariances between cross-sections. The difficulty with missing values or non-rectangular data sets is that each element of E'E may not in fact be based on fully T pairs of observations. My procedure solves this by substituting zeros for missing values in X and E and rectangularizing both also with zeros, and dividing each element of E'E by the number of (pairs of) observations on

[4] *I.e.*, toward the middle rather than toward the end of a CS's data.

which it is actually based--*i.e.*, not counting the pairs where one or the other (or both) observation(s) is (are) missing. This is done by creating another TxN matrix, M, where each element is equal to one if the corresponding observation in E is not missing and zero if it is. Then, E'E divided element-by-element by M'M may replace E'E/T in the above formula and all is correct.

The procedure has been tested (in Gauss 386-i v3.1 and 3.2.13 running under DOS and a DOS Window under Windows 3.1) as follows. First, I created some artificial data, rectangular and without missing values. X, the independent variable is random normal. e, the true error is also random normal and independent of X. Y, the dependent variable, was set equal to X+e. Beck's GAUSS PCSE procedure (thank you to Neal Beck for the code) was then run on these data. In this sample, my procedure produces identical results. This proves only that my procedure works in perfect rectangular data sets if Beck's does, but I consider that about as safe a bet as can be found. Testing how my procedure compares to Beck's in non-rectangular data-sets and/or those with missing values cannot be done because the latter cannot deal with such data to provide the baseline. I did perform a very crude test in this setting by, again generating random normal and independent data as above, only non-rectangular this time. The results are generally close to equal to the OLS results, and converge to it in larger samples, as we would expect if the procedure is working correctly. I fear that is the best that can be done by way of a comparison to the ideal.

More generally, the procedure seems to operate bug-free. (I have been using it extensively in a data set with 850+/- rows, but only 550+/- usable observations; the data are non-rectangular and have a few internal missing values). It has also been used/tested this fall by Gary King's second-semester graduate statistics class without mishap following one important correction (thank you to Ken Scheve for noticing it).

Highlights of what the procedure does:

(1) Produce PCSE's in any data set where they are theoretically defined.

(2) Automatically generate and use CS and TP fixed-effect dummies if requested.[5]

(3) Estimate and incorporate Panel WLS information prior to applying PCSE's if requested.

> N.b., this is done in two stages. First, OLS is estimated. The residuals from that are squared and regressed on the CS indicators (less one of them) and a constant. The F-test from this regression can be seen as a test of homoskedasticity as the null against CS-specific variance (and the DW from this auxiliary regression can be seen as a test of ARCH(1) against the null of panel heteroskedasticity). The inverse of the square root of the fitted values from this auxiliary regression are then used as weights to transform the original data. OLS is run on the transformed data, which produces the Panel WLS results. The PCSE estimation is then applied to these results. The line in the procedure where the squared residuals are regressed on the CS indicators and related

[5] See Smith (1995) for a cogent discussion of the substantive interpretation of models with each sort of fixed-effects.

commands can be changed to a different model of error-variance if the researcher desires with minimal hard-wiring. They are flagged for easy recognition.

(4) GAUSS's *OLS.SRC* procedure produces incorrect Durbin-Watson statistics when there are missing values in the middle of the data (Aptech has been notified). The problem is that OLS removes missing values from the data eventually passed to the part of the procedure where estimates are made. This means that later, when any time-series properties (statistics) are analyzed (computed) some of the adjacent observations are no longer temporally adjacent. This implies, for example, that the Durbin-Watson statistic is not based on the correct ordered set of residuals. This problem is fixed prior to the reporting of DW-stats in output from my procedure, but it remains in OLS.SRC.

The text of the procedure (*PCSE.G*) follows. It can also be downloaded from the American Unive2rsity GAUSS Procedure Archive along with more detailed instructions on how to use it than are incorporated as comments into the procedure itself.

Text of the file PCSE.G :

```
@ Gauss Procedure to Estimate Linear Regression, Implementing
* Panel-Corrected Standard-Errors (PCSE) (Beck and Katz (APSR 1995, 1997))
* in the Presence of Non-rectangular and/or Missing Data
*
* Please cite "PCSE.G: A Gauss Procedure to Implement Panel-Corrected
* Standard-Errors in Non-Rectangular Data Sets."
*
* by Robert J. Franzese, Jr. (Harvard University) October 1995
* Revised 3/19/96
*
* Code is written and submitted for public, non-commercial use.
* The authors assumes no responsibility whatsoever for the
* performance of the procedure.
*
* Procedure handles non-rectangular data and/or missing observations,
* and (optional) creates and uses cross-section and/or time-period
* fixed effects, and (optional) estimates and uses panel weights (other
* linear variance models can be estimated if desired by editing (hardwiring)
* the line `vx = ...' (remarked in the procedure) A non-linear model could
* of course be substituted.
*
* AUTHOR'S NOTE:
* The assistance of Curt Signorino (Harvard University) in optimizing
* the procedure to reduce computer time and memory use is gratefully
* acknowledged. Gratitude also to Ken Scheve for discovering an error
* in the earlier version.
*
* The author was funded by a Mellon Foundation Dissertation
* Completion Grant during the writing of the procedure.
*
* FORMAT:
* {m,b,stb,zz,stderr,se,sigma,cx,rsq,rbsq,resid,dwstat,nobs,degfree,
* tstats,poft,ursq,urbsq,usigma,udw,ures,corre}
* =pcse(vnames,cnames,y,x,nocs,tstart,tend,weight,cdum,tdum);
*
* Procedure assumes data organized {[c1,t1],[c1,t2],...,c[2,t1],...,etc.}',
* that each cross-section has the same number of rows of the data matrix
* allotted to it, though some of those rows may be missing data, and that
* the first column of "indvars" is the constant. (Thus, if your data is non-
* rectangular, you must rectangularize it by putting missing values in. Any
* observations you wish to drop must be fed as missing in the dep var or at
* least one indep var.)
*
* Procedure returns: 22 total, in the following order:
 * m = (matrix) moment matrix (OLS/WLS)
 * b = (vector) coefficients (OLS/WLS)
 * stb = (vector) standardized coefficients (OLS/WLS)
 * zz = (matrix) variance-covariance of coefficients (OLS/WLS + PCSE)
 * stderr = (vector) coefficient standard-errors (OLS/WLS)
 * se = (vector) coefficient standard-errors (OLS/WLS + PCSE)
 * sigma = (scalar) std err of the estimate (WLS, same as usigma if OLS)
 * cx = (matrix) variance-covariance of X and Y (OLS/WLS)
 * rsq = (scalar) R-squared (WLS, same as ursq if OLS)
 * rbsq = (scalar) R-bar-squared (WLS, same as urbsq if OLS)
 * resid = (vector) residuals (WLS, same as ures if OLS)
 * dwstat = (scalar) Durbin-Watson statistic (WLS, same as udw if OLS)
 * nobs = (scalar) number of valid observations
 * degfree = (scalar) degrees of freedom of coefficient estimates
 * tstats = (vector) t-stats of test that coeffs=0 (OLS/WLS + PCSE)
 * poft = (vector) 2-sided prob of those t-tests (OLS/WLS + PCSE)
 * ursq = (scalar) R-squared (unweighted residuals used)
 * urbsq = (scalar) R-bar-squared (unweighted residuals used)
 * usigma = (scalar) std error of estimate (unweighted residuals used)
 * udw = (scalar) Durbin-Watson statistic (unweighted residuals used)
 * ures = (vector) unweighted residuals
 * corre = (matrix) Cross-section Correlation of Residuals

* Procedure Inputs: 10 in the following order:
 * vnames = character vector: variable names, last being dep var
 * cnames = character vector: cross-section unit names
 * y = vector: observations on dependent variable
 * x = matrix: obs. on independent variables including constant (first)
 * time and or c-s dummies can be added for you, omit them here
 * nocs = scalar: number of cross-sectional units
 * tstart = scalar: first time period (earliest from all c-s units)
 * tend = scalar: last time period (latest from all c-s units)
 * weight = scalar: 1 for WLS, 0 for OLS
 * default WLS model is panel weights, can change that directly by
 * hard-wiring the line `vx = ...' in this procedure
 * cdum = scalar: 1 = add c-s dummies, 0 = no c-s dummies
 * tdum = scalar: 1 = add time dummies, 0 = no yime dummies
 * if tdum=1 and cdum=1, procedure drops 1st time dummy @

proc (22) =
 pcse(vnames,cnames,y,x,nocs,tstart,tend,weight,cdum,tdum);

local i,j,t,cdums,tdums,denom,vcve,corre,zz,cm,valid,nindvar,nobs,
 wx,wy,vnam,m,b,stb,vc,stderr,sigma,cx,rsq,resid,dwstat,se,degfree,
 tstats,poft,results,mask,fmt,ures,ursq,urbsq,usigma,udw,tv,rbsq,
 sse,usse,nots,res2,vx,vare,minvar,wvare,wtv,res,resm,uresm,xm,ym,
 wxm,wym,tseq,rresm,rvalid,diagvcve,uresm1l,validm1l,resm1l;

/*declare vars local to procedure*/

format 12,6; /* resets default format for screen output */

nots = tend - tstart + 1; /* computes number of time periods */

if rows(y) /= rows(x); /* this set of ifs checks if input valid */
 "ERROR: Different number of rows in X and Y";
 end;
elseif rows(y) /= nocs*nots;
 "ERROR: Rows in Y and X do not equal # of cs times # of t-p";
 end;
elseif rows(cnames) /= nocs;
 "ERROR: Number of c-s names does not equal number of c-s";
 end;
elseif rows(vnames) /= cols(x)+1;
 "ERROR: Number of variable names does not equal columns X + columns Y";
 end;
else; /* no errors in input detected, create an indicator for valid obs */
 valid = y~x; /*make valid the data matrix */
 valid = (valid./=miss(1,1)); /*set each element to 1 if not miss 0 else*/
 valid = prodc(valid'); /*creates column vector with 1 if valid obs */
 validm1l = missrv(lag1(valid),0); /* lag of valid */
 y = valid.*y+(1-valid).*"miss"; /*put "miss" in y for all missing*/
 y = miss(y,"miss") ; /* convert "miss" to missing value code */
 x = valid.*x+(1-valid).*("miss".*ones(1,cols(x))); /* ditto for x */
 x = miss(x,"miss") ;
endif;

/** Describing of Data Set Shape **/
nobs=sumc(valid); /* compute number of valid observations */
print ftos(nots,"Maximum Number of Time-Periods = %*.*lf",0,0);
print ftos(nocs,"Maximum Number of Cross-Sections = %*.*lf",0,0);
print ftos(nots*nocs,"NoT x NoC (possible observations) = %*.*lf",0,0);
print ftos(nobs,"Actual Number of Observations Valid = %*.*lf",0,0);
print ftos(nots*nocs-nobs,"Number of Invalid Observations = %*.*lf",0,0);

if cdum==1; /* create a matrix of c-s dummies if requested */
 /* delete constant and adjust variable names to match */
 vnames = vnames[2:rows(vnames),.]; /* delete var name for constant*/
 vnames = cnames|vnames; /* append c-s names to remaining var names */
 cdums = (eye(nocs).*.ones(nots,1)); /* create matrix of c-s dummies */
 cdums = valid.*cdums+(1-valid).*("miss".*ones(1,nocs)); /*put miss in*/
 cdums = miss(cdums,"miss");
 x=x[.,2:cols(x)]; /* remove constant from x matrix */
 x=cdums~x; /* c-s dummies are put at front of x-matrix */
elseif cdum /= 0;
 "ERROR: `cdum' argument must be a scalar, 0 (off) or 1 (on)";
 end;
endif;

/** generate time-period dummies **/

if tdum==1; /* create a matrix of time dummies if requested */
 /* delete constant if cdum hasn't already done so */
 /* append time dummies matrix to front of x and */
 /* adjust vnames to match the resulting x matrix */
 tdums = (ones(nocs,1).*.eye(nots)); /* create matrix of t-p dummies */
 tdums = valid.*tdums+(1-valid).*("miss".*ones(1,nots)); /* put miss in */
 tdums = miss(tdums,"miss");
 tseq = seqa(tstart,1,(tend-tstart+1));
 if cdum == 0 ;
 vnames = vnames[2:rows(vnames),.]; /* remove constant name */
 x = x[.,2:cols(x)]; /* remove constant */
 vnames = (0$+"T"$+ftocv(tseq,0,0))|vnames; /*append tnames to v-names*/
 x = tdums~x; /* append tdums to x-matrix */
 else ; /* if c-s dums, same, except trim first time-period dummy */
 vnames = trimr((0$+"T"$+ftocv(tseq,0,0)),1,0)|vnames;
 x = (trimr(tdums',1,0))'~x;
 endif;
elseif tdum /=0;
```

```
 "ERROR: `tdum' argument must be a scalar, 0 (off) or 1 (on)";
 end;
endif;

nindvar=cols(x); /* now compute total number of independent variables */
print ftos(nindvar,"Number of Independent Variables = %*.*lf",0,0);
"";
degfree = nobs-nindvar; /* compute degrees of freedom */

if weight==0; /* if weighting not requested, do OLS */
 "Using OLS in First and Second Stage...";
 "";
 __altnam = vnames; /* feeds vnames to OLS */
 __miss = 1; /* instructs OLS to omit missing data */
 __con = 0; /* instructs OLS not to add another constant */
 _olsres = 1; /* instructs OLS to compute residuals and DW-stat */
 __output = 0; /* set this to 1 if you want OLS results printed out */

{ vnam,m,b,stb,vc,stderr,sigma,cx,rsq,resid,dwstat }=OLS(0,y,x); /* OLS */

 rbsq = 1-(1-rsq)*((nobs-1)/degfree); /* compute r-bar-squared */
 res = y-x*b; /* compute nocs*nots by 1 vector of residuals */
 resm = missrv(res,0); /* an alternate resid vector with 0 for missing*/
 resm1l = missrv(lag1(resm),0); /* lag of resm */
 ures=res; /* unweighted residuals, same as OLS here */
 uresm = missrv(ures,0); /* alternate with zeros for missings */
 sse=resm'resm; /* compute sum of squared errors */
 usse=sse; /* unweighted sum of squared errors, same as OLS here */
 ursq=rsq; /* unweighted r-squared, same as OLS here */
 urbsq=rbsq; /* unweighted r-bar-squared, same as OLS here */
 usigma=sigma; /* unweighted std err of est, same as OLS here */

/** nb: GAUSS OLS computes incorrect DW when missing values present **/

 denom = sumc(valid.*validm1l.*resm^2); /* denom for DW is sum sq res */
 dwstat = sumc(valid.*validm1l.*(resm-resm1l)^2); /* numerator is... */
 dwstat = dwstat/denom; /*...sum of squared differences */
 udw=dwstat; /* unweighted DW-stat, same as OLS here */

 "First-stage done, now computing Panel V-Cov of Residuals...";

 @ covariance of resid[i,t] with resid[j,t] is computed
 * taking care to divide sum of cross-products only by the
 * number of valid observations @
 rvalid = reshape(valid,nocs,nots)'; /* reshape valid to nots x nocs */
 rresm = reshape(resm,nocs,nots)'; /* ditto for residual vector */
 vcve = rresm'*rresm; /* generate nocs by nocs matrix of crossprods */
 denom = rvalid'*rvalid;
 denom = miss(denom,0);
 if ismiss(denom);
 "ERROR: A c-s unit exists w/o any observations or w/o "\
 "any observations in common with another c-s unit.";
 "You must remove that c-s unit from the data passed to PCSE.";
 end;
 endif;
 vcve = vcve./denom;

 "Done, now computing cross-section residual-correlation matrix...";

 diagvcve = sqrt(diag(vcve));
 corre = vcve./(diagvcve*diagvcve');

 "Done, now computing PCSE V-COV of b...";
 xm = missrv(x,0); /* GAUSS doesn't like to do m-algebra with missing */
 ym = missrv(y,0); /* obs, oblige it by replacing with 0. This has */
 /* no effect on x'x, just watch the denominators */
 tv = ym'ym-nobs*((sumc(ym)/nobs)^2); /* sum of squared variance of y */
 wtv = tv; /* sum of squared variance of weighted y, same for OLS */

 zz = (inv(xm'xm))*xm'(vcve.*.(eye(nots)))*xm*(inv(xm'xm));
 /*PCV-COV(b)*/

 se = sqrt(diag(zz)); /* PCSE's */
 tstats = b./se; /* PC t-stats */
 poft = 2*cdftc(abs(tstats),degfree); /* 2-sided p of |PCSE t's| */

elseif weight==1; /* if weighted least squares requested */
 "Using OLS in First Stage...";
 "";
 __altnam = vnames; /* feed OLS the relevant variable names */
 __miss = 1; /* instruct OLS to omit missings */
 __con = 0; /* instruct OLS not to add another constant */
 _olsres = 1; /* instruct OLS to compute residuals and DW-stat */
 __output = 0; /* change this to 1 if you want the OLS output */

{ vnam,m,b,stb,vc,stderr,sigma,cx,rsq,resid,dwstat }=OLS(0,y,x); /* OLS */

"Done, now calculating weighting series using OLS on OLS residuals squared";
```

```
 res2=(y-x*b).*(y-x*b); /* compute nocs*nots by 1 vector of residuals */
 if cdum==0; /* check to see if matrix of c-s dummies exists, if not...*/
 cdums = (eye(nocs).*.ones(nots,1)); /* create matrix of c-s dums */
 cdums = valid.*cdums+(1-valid).*("miss".*ones(1,nocs));
 cdums = miss(cdums,"miss");
 endif;

@ Residual Variance Model can be changed by editing the line `vx = ...' @

 vx = ones(nocs*nots,1)~cdums[.,2:nocs] ; /* ind vars in res var model */

@ The following process creates a weighting series by regressing the squares
* of the OLS residuals on the cross-section dummies (and whatever other
* explanitory variables you might have included in the preceding line.)
* The inverse of the square roots of the fitted values are the weights.
* Without other explanitors than the c-s dummies, this is exactly equivalent
* to panel-weighted least-squares @

 "----> n.b., F-stat is test of variance model against homoskedasticity";
 "";

@ if `vx= ...' is changed, the following line must be altered to match @

 __altnam = "one"|cnames[2:nocs,1]|"SqOLSres";
 __output = 1; /* 0 if you don't want the res var model output */

 { vnam,m,b,stb,vc,stderr,sigma,cx,rsq,resid,dwstat }=OLS(0,res2,vx);

 vare = vx*b; /* make nocs*nots by 1 vector of fitted res variance */

@ The following if...endif checks for negative fitted values
* in the variance model, replacing them with the minimum positive value.
* (author's note: substitution of a more appropriate model than a linear
* model of variance may be particularly appropriate if negative fitted
* values are common) They are not possible in the panel-weight set-up. @

"";
"Calculating Weighting Series..."; /* Set negative wts to min positive */
 if minc(vare) <= 0;
 minvar = minc((vare.>0).*vare+(vare.<=0).*maxc(vare));
 vare = (vare.>0).*vare+(vare.<=0).*minvar;
 endif;

 wvare = ones(rows(vare),1)./sqrt(vare); /* weighting series */
 wx = wvare.*x; /* transform x to weighted x */
 wy = wvare.*y; /* transform y to weighted y */
 wxm = missrv(wx,0); /* create alternates wx and wy where miss are 0 */
 wym = missrv(wy,0);

"Done, now doing WLS using OLS on transformed variables...";

 __altnam = vnames;
 __output = 0; /* change this to 1 if you want WLS output */

{ vnam,m,b,stb,vc,stderr,sigma,cx,rsq,resid,dwstat }=OLS(0,wy,wx); /*WLS*/

 res = wy-wx*b; /* nocs*nots vector of weighted residuals */
 resm = missrv(res,0); /* alternate with 0 for missing */
 resm1l = missrv(lag1(resm),0); /* lag of resm */
 rbsq = 1-(1-rsq)*((nobs-1)/degfree); /* weighted r-bar-squared */
 ures = y-x*b; /* calculate unweighted residuals */
 uresm = missrv(ures,0); /* alternate, 0 for missings */
 ym = missrv(y,0); /* alternate y with zeros for missings */
 tv = ym'ym-nobs*((sumc(ym)/nobs)^2); /* unweighted sum of var(y) */
 wtv = wym'wym-nobs*((sumc(wym)/nobs)^2); /* weighted sum of var(y) */
 ursq = (tv - uresm'uresm)/tv; /* unweighted r-squared */
 urbsq = 1-(1-ursq)*((nobs-1)/degfree); /* unweighted r-bar-squared */
 sse=resm'resm; /* sum of wwighted residuals variance */
 usse = uresm'uresm; /* unweighted sum of squared residuals */
 /* n.b.,in WLS, unweighted resids may not have mean 0 */
 usigma = sqrt((uresm'uresm)/degfree); /* unwtd std err of estimate */
           /* computing unweighted DW-stat */
 uresm1l = lag1(uresm); /* create lagged residual (alternate) */
 uresm1l = missrv(uresm1l,0); /* replace missings with zeros */
 validm1l = missrv(lag1(valid),0); /* ditto for valid */
 denom = sumc(valid.*validm1l.*uresm^2); /* denom for DW is sum sq res */
 udw = sumc(valid.*validm1l.*(uresm-uresm1l)^2); /* numerator is... */
 udw = udw/denom; /*...sum of squared differences */

/** nb: GAUSS OLS computes incorrect DW when missing values present **/

 denom = sumc(valid.*validm1l.*resm^2); /* denom for DW is sum sq res */
 dwstat = sumc(valid.*validm1l.*(resm-resm1l)^2); /* numerator is... */
 dwstat = dwstat/denom; /*...sum of squared differences */

"Done, computing PCSE V-COV of WLS residuals...";
 rvalid = reshape(valid,nocs,nots)'; /* reshape valid to nots by nocs */
 rresm = reshape(resm,nocs,nots)'; /* ditto for resids */
 vcve = rresm'*rresm; /* calculate nocs by nocs cross-prod of resids */
 denom = rvalid'*rvalid; /* calculate nobs for each element of vcve */
 denom = miss(denom,0);
 if ismiss(denom);
 "ERROR: A c-s unit exists w/o any observations or w/o "\
 "any observations in common with another c-s unit.";
 "You must remove that c-s unit from the data passed to PCSE.";
 end;
 endif;
 vcve = vcve./denom; /* cross-prods over nobs for each is covar matrix */

"Done, computing cross-section residual-correlation matrix...";

 diagvcve = sqrt(diag(vcve)); /* sq root of diag elements = std dev */
 corre = vcve./(diagvcve*diagvcve'); /* vcv over sd()sd() = corr */

"Done, computing PCSE V-COV of b..."; /* PCSE V-COV(b) with wtd resids */

 zz = (inv(wxm'*wxm))*wxm'*(vcve.*.(eye(nots)))*wxm*(inv(wxm'*wxm));
            /* PCV-COV */

 se = sqrt(diag(zz)); /* PCSE's from WLS */
 tstats=b./se; /* PC t-stats from WLS */
 poft = 2*cdftc(abs(tstats),degfree); /* 2-sided p of |PC t| */

else;

 "Weight parameter must be set as 0 (=off) or 1 (=on)";
 end;

endif;
"** PCSE Results **"; /* printing out the results pretty-like */
"";
print ftos(nobs,"Valid Cases: %*.*lf",20,0);;
print ftos(vnames[rows(vnames)]," Dependent Variable:%*.*s",20,0);
print ftos(nocs*nots-nobs,"Missing Cases:%*.*lf",20,0);;
print " Deletion Method: ";;
print "Listwise"; /** always uses listwise **/
if weight == 1;
 print ftos(wtv,"Wtd Total SS: %*.*lf",20,3);;
 print ftos(degfree," Degrees of Freedom:%*.*lf",20,0);
endif;
print ftos(tv,"Total SS: %*.*lf",20,3);;
print ftos(degfree," Degrees of Freedom:%*.*lf",20,0);
if weight == 1;
 print ftos(rsq,"Wtd R-Squared:%*.*lf",20,3);;
 print ftos(rbsq," Wtd Rbar-Squared: %*.*lf",20,3);
```

```
endif;
print ftos(ursq,"R-Squared: %*.*lf",20,3);;
print ftos(urbsq," Rbar-Squared: %*.*lf",20,3);
if weight == 1;
 print ftos(sse,"W Residual SS:%*.*lf",20,3);;
 print ftos(sigma," Wtd Std Error Est: %*.*lf",20,3);
endif;
print ftos(usse,"Residual SS: %*.*lf",20,3);;
print ftos(usigma," Std Error of Est: %*.*lf",20,3);
print ftos(udw,"DW-Stat: %*.*lf",20,3);;
print ftos(dwstat," Wtd DW-Stat: %*.*lf",20,3);
print;
print " PCSE Std Prob Sta"\
"ndardized Reg Std";
print "Variable Estimate Error t-value >|t| E"\
"stimate Error";
print "------------------------------------------------------------"\
"-----------------";
results = vnames[1:nindvar,1]~b~se~tstats~poft~stb~stderr;
mask = 0~1~1~1~1~1~1;
let fmt[7,3] = "-*.*s" 9 8 "*.*lf" 12 6 "*.*lf" 12 6 "*.*lf" 12 6 ""\
 "*.*lf" 10 4 "*.*lf" 12 6 "*.*lf" 12 6;
call printfm(results,mask,fmt);
"";

retp(m,b,stb,zz,stderr,se,sigma,cx,rsq,rbsq,res,dwstat,
 nobs,degfree,tstats,poft,ursq,urbsq,usigma,udw,ures,corre);

endp;
```

MA5: An Example of an E-Views Program to Estimate Cross-Validated Standard-Errors

Beck and Katz (1993) have pointed out the utility of *cross-validation* of residuals as a basis model selection and evaluation. Simply, cross-validation is the process of re-estimating an equation several times, each time leaving some portion of the sample out. (The sets left out in each case should be exclusive and jointly cover the whole sample.) The results from each of these auxiliary regressions are then used to *predict* the omitted portion. Each time the prediction errors are saved. The standard deviation of the predictions is called the cross-validated standard-error of the regression, *s*.

Using *s* for model selection, then, we will be making our choices less based on how a model *fits* a particular data set, but in how well it can predict outcomes not included in estimating it. It is straight-forward to note the particular utility of this procedure in analyzing pooled time-series-cross-section data since it is has particular substantive meaning if we omit a cross-section each time. Specifically, we are seeing how well using the experience of the other countries helps us to predict what should happen in the omitted country given its independent variables. Thinking back to the goal of theory that *travels usefully* discussed in Chapter I, then, it is clear that *s* can also be interpreted as an answer to the question: "How well do our theories travel?"

An example of an E-views program (batch file) which estimates *s* from a data set stacked vertically by country, *i.e.*, ([c1,t1],[c1,t2],...,[c1,tT],[c2,t1],...,[c2,tT],...,[cN,t1],...[cN,tT])′ follows. The example is from Chapter IV.[6]

[6] I intend to translate to an automated GAUSS for implementation in Chapters II-III and future work.

CH4XVAL.PRG

```
SMPL 1 84                                       Sample size ignoring missing data was 84 (21*4)
                                                         (i.e., 21 countries, 4 decades)
GENR ERR = NA                                   Initiates the cross-validated errors variable

FOR !A = 1 TO 21                                !A is a MicroTSP/E-Views counter

 IF !A <> 12 THEN                               Greece is country 12; no data on GOV for Greece; to
                                                prevent an iteration with 0 observations skip !A=12

 SMPL 1 84                                      Reset the sample each iteration
 SMPL IF CTRY<>!A                               Exclude country !A from the sample
 LS(W=WT,H) LNUE C D70 D80 LNUE1L LNUE1L2
            LNUE1L3 COV CWB MANMQ LGOV          Run the regression in question (this is Model I)
CBI
            OIL70 OIL70(-1)                     Reset the sample
 SMPL 1 84                                      Set sample to only country !A
 SMPL IF CTRY=!A
 GENR ERR = LNUE -C(1) -C(2)*D70 -C(3)*D80
            -C(4)*LNUE1L -C(5)*LNUE1L2          Generate the prediction error in country !A from the
            -C(6)*LNUE1L3    -C(7)*COV          equation just estimated without country !A
-C(8)*CWB
            -C(9)*MANMQ -C(10)*LGOV
            -C(11)*CBI-C(12)*OIL70              End the "if" loop regarding Greece
            -C(13)*OIL70(-1)                    Do another iteration
 ENDIF                                          Reset the sample
NEXT
SMPL 1 84                                       [then record the standard deviation of ERR]
```