Let me say at the outset that three caveats must be made about this paper. First, I am not an empirical economist (much less an empirical macro economist) in most senses of the term because I do not publish empirical work. Hence most of what I say here comes from consulting with students and faculty, or from my avocation of fooling around with economic models and computers. Second, I am described as a knee jerk SAS freak--translated, I only program in most of what I say here comes from consulting with students and faculty, or from my avocation of fooling around with economic models and computers. Second, I am described as a knee jerk SAS freak. The third caveat is that Phil Miller attempted to ask SAS to let me remain a knee jerk SAS freak. The third caveat is that Phil Miller suggested that I write this paper, and I might not have without his asking--although I think he wanted "what could be" and not "what isn't". But we users of SAS must complain sometime in order to improve the product.

For both parts of this paper, I must say that SAS's motto is that SAS SAVES TIME. If it does for we would be COBOL programmer's doing data management, then it also should for the econometrician who would use PEC, TSP, ESP, TROLL, SHAZAM or some other package. SAS does not save time in some places but does not in others. What I would hope to accomplish here is to describe some of my troubles, some of my desires, and to get the users to demand more from SAS.

Now for this first part of the paper: What Isn't- I contend that the users of ETS deserve better than what they have. I view the problem as a lack of programming time, and the type of programmer. The ETS package probably generates $500,000 revenue or more to SAS but only Dave DeLong and John Sell program on most of the procedures, and then only a part of their time. Neither is an empirical researcher, and this biases how the programs are written and what 'bells and whistles' are given to the user. I think we deserve two or maybe even three full time programmers/econometricians to support the package. When you look at ESP, TSP, TROLL or SHAZAM and what can be done in them, you can see the econometrician's influence. And this influence is important to how useful and how time saving the package is. It takes both a good programmer and the design consideration of the practicing empirical econometrician to make the mix excellent. But enough said about how SAS should spend its dollars.

The problems that I will discuss fall into three general categories--bugs in current procedures, weaknesses in current procedures and procedures which do not exist. The first major problem with SAS/ETS for the general user is the number of bugs that survive SAS, alpha and beta testing. The problem is that to the general user, faith is lost whenever a bug is found. A bug is found, you must make any model with a dependent variable lagged more than one period. For example, in an equation such as

\[
Y = A0 + A1X + A2LAG(Y) + A3LAG2(Y)
\]

a static simulation is incorrect in SAS/ETS 79.6. The value used for LAG2(Y) in the simulation is in reality LAG(Y)-in fact LAG(Y) is used for any LAGk(Y) in the static simulation. The dynamic simulation "works" correctly, but since static simulation is used to begin a dynamic simulation, one does not get a correct dynamic simulation from this equation.

The only way is to fool SAS in as follows:

\[
Y = A0 + A1X + A2LAG(Y) + A3LAG2(YLAG) + A4LAG(YLAG).
\]

That is, we make a two equation model from one so that we do not have any lag greater than 1. In order to deal with longer lags, you must add more equations. Although this works, it is cumbersome when an equation has twenty or so lags, which is very common in empirical macro models. We now write models using this form for all of our work to be sure that we circumvent the simulation bug. The point is that the bug should not have survived to the production release.

Errors of this type-numbers are produced which are wrong yet believable-are difficult to detect. We spend about 3 weeks constructing a sure fire model and data that would produce the error. I feel that SAS, and not the user, should be responsible to devise methods to detect errors like this, and then to fix them before a production release. If this requires more programmer time, then so be it. It is far better to have fewer procedures (in the production release) that can be trusted than many procedures which can not.

The second major problem (weakness) with SAS and SAS/ETS for the econometrician is that there are a lot of special purpose procedures which do one thing well, but not two or more things combined. As an example of this problem (and possibly the most important one) PROC AUTOREG is an exceptionally nice procedure for estimating a single equation with autocorrelation. It does a true GLS, and will now iterate so that the Cochran-Orcutt iterative
First, there are many consistent ways to estimate the first autocorrelation parameter (neglect higher orders for the moment). SAS uses only one of them so that you are forced to their method whether or not you like it (much less can determine what it is from the documentation). It would be nice to be able to specify the method (from the many possible-Prais Winston, Durbin, regression coefficient of the residual on the residual lagged, etc.) used to calculate this parameter. At least the LAG statement could be made to act like a RESTRICT statement, restricting the autocorrelation coefficients to specified values rather than just 0 as is now allowed. This could be achieved making the LAG statement act like a restrict statement on the autocorrelation coefficients. At least then one could determine the estimates of the autocorrelation coefficients that he wants to be used separately, and then run AUTOREG with a LAG statement like LAG 1=.75 3=.45. As it is now, we are stuck with the estimate that SAS calculates.

A much more difficult problem to circumvent is estimation of an autoregressive equation in which there is an "endogenous" variable. By endogenous variable, I mean a variable on the right side of the equation that theoretically must be correlated with the error of the equation. This could be a current period endogenous variable or a lagged endogenous, or an errored variable. All three of these problems require an instrumental variable technique for estimation.

Let me be more explicit with an example of a lagged endogenous variable appearing on the right hand side of the equation.

\[ Y = AO + A1X + A2*LAGY + E \]
\[ E = RH01*ELAG + U \]

where \( U \) is nice Gaussian white noise. Since both \( E \) and \( LAGY \) involve \( ELAG \), a right hand side variable is correlated with equation's error term. Direct estimation - either by SYSSREG or by AUTOREG with \( XLAG=1 \) of this equation results in inconsistent, biased estimates.

One suggested technique for this problem is to estimate the parameters at the first stage (to determine residuals which then are used to determine an estimate for RHO) by running the regression

\[ Y = AO + A1X + A2*WHATLAG \]

where \( WHATLAG \) is an instrument for \( YHAT \), obtained as the predicted from another equation. In fact SAS documentation for AUTOREG shows the following example:

```
PROC SYSSREG DATA=A OUT=B;
  MODEL Y=XLAG;
  OUTPUT P=YHAT;
```

```
DATA C; WHATLAG=YHAT;
  SET B; RETAIN WHATLAG;
  PROC AUTOREG DATA=C;
    MODEL Y=WHATLAG X /XLAG=1;
  (pg 133 SAS USERS GUIDE '79)
```

Unfortunately, estimating in this way produces inconsistent results since \( WHATLAG \) serves as a proxy variable for \( XLAG \) in AUTOREG rather than as an instrument. The difference is in the calculation of the residual which of course is used to calculate the first order autoregressive coefficient. AUTOREG calculates the residual as

\[ EWRONG = Y - AO - A1X - A2*WHATLAG \]

where \( AO, A1 \) and \( A2 \) are of course the estimated coefficients. But it should be calculated (for consistent estimates of RHO anyway) as

\[ ERIGHT = Y - AO - A1X - A2*YLAG \]

I.e., the true value of \( YLAG \) must be used rather than \( WHATLAG \). Unfortunately, AUTOREG can not be fooled into this method no matter what, since it will use the proxy variable \( WHATLAG \) to calculate the residual, which results in an inconsistently estimated autoregressive parameter.

The only way around the problem is to independently estimate the first stage equation with REG or SYSSREG, calculate the residuals and an estimate of RHO in a data step, calculate weighted first differenced variables and then estimate again on the weighted first differenced variables. Instead of the PROC AUTOREG step above we would need the following UNTESTED code which is the only approach in SAS I know (unless one resorts to PROC MATRIX) is:

```
PROC REG DATA=C OUTEST=REGOUT;
  MODEL Y = WHATLAG X;
DATA B; SET C; WHATLAG=YHAT;
  PROC SCORE DATA=D SCORE=REGOUT;
    VAR Y WHATLAG X;
    DATA E; SET RESIDS TYPE=OLS;
      CROSS*Y*LAG(Y); KEEP RHO;
      VAR+Y*Y;
    IF EOF THEN DO;
      END=
    DATA F; SET E;
    IF EOF=0;
      DO UNTIL (EOF);
      SET B END=EOF;
      Y=Y-HLAG*RHO;
      DX=X-HLAG*RHO;
      DYLAG=YLAG-YLAG*HLAG*RHO;
      IF DYLAG=. AND DX NE'. THEN DO;
        DX=SQRT(1-RHO*RHO);
        DYLAG=SQRT(1-RHO*RHO);
        IF DYLAG NLAG AND Y NLAG THEN DO;
          NLAG=DYLAG; NLAG=SQRT(1-RHO*RHO);
          END;
    DATA E; SET RESTDS TYPE=OLS;
      CROSS*Y*LAG(Y); KEEP RHO;
      VAR+Y;"'
    IF EOF THEN DO;
      END=
    DATA F; SET E;
    IF EOF=0;
      DO UNTIL (EOF);
      SET B END=EOF;
      DLAG=HLAG*RHO;
      IF DLAG=. AND Y NE . THEN DO;
        DLAG=HLAG; DLAG=SQRT(1-RHO*RHO);
        END=
    DATA E; SET RESTDS TYPE=OLS;
      CROSS*Y*LAG(Y); KEEP RHO;
      VAR+Y;"'
    IF EOF THEN DO;
      END=
    DATA F; SET E;
    IF EOF=0;
      DO UNTIL (EOF);
      SET B END=EOF;
      DLAG=HLAG*RHO;
      IF DLAG=. AND Y NE . THEN DO;
        DLAG=HLAG; DLAG=SQRT(1-RHO*RHO);
        END=
    DATA E; SET RESTDS TYPE=OLS;
      CROSS*Y*LAG(Y); KEEP RHO;
      VAR+Y;"'
    IF EOF THEN DO;
      END=
    DATA F; SET E;
    IF EOF=0;
      DO UNTIL (EOF);
      SET B END=EOF;
      DLAG=HLAG*RHO;
      IF DLAG=. AND Y NE . THEN DO;
        DLAG=HLAG; DLAG=SQRT(1-RHO*RHO);
        END=
```

This is not a time saving way to estimate, especially if there is any specification search to be done.
Add to this problem the lack of more modern techniques such as those in Fair(1970), or Cooper(1972), or a maximum likelihood technique (Beach and MacKinnon(1978)) which are available in TSP and others, and AUTOREG loses its prowess for even knee jers like me. What is needed is to construct procedures that either do it all, or that can be interfaced to do it all. As it stands, AUTOREG serves only those rare instances of single equations that do not have contemporaneously correlated right hand side variables, and of course have no need of the abilities of PROC REG for diagnosis. It's a shame from the user point of view that SYSREG was not given both REG's and AUTOREG's capabilities to estimate and diagnose models within a system framework, rather than creating new procedures to do one special thing.

Another sort of weakness in SAS is the lack of documentation about the statistical procedures. An example of the difficulty is estimating a system in PROC MODEL and SYSLIN in which you have lagged exogenous variables, but use the LAG() function to reference them—i.e., LAG(Y) rather than XLAG. In such a case, the variable is not placed in the exogenous list, as it would and should be if you used XLAG. This of course results in bad estimates since you do not have the 'correct' set of instrumental variables for the first stage regressions. Looking at the documentation does nothing to help the general econometrician-in fact there is no warning at all that this is the case. In PROC MODEL for exogenous variables since it can not be used on the EXOGENOUS statement.

Some warning is given in the manual that IF...THEN...ELSE statements should not be used for estimation, but none is given for simulation. Any of you daring enough to have tried to simulate a system with IF...THEN...ELSE statements will know that depending on which option (NEWTON,SEIDEL, or JACCOBI) is used, these statements will have different effects on the simulation since they are executed differently. In fact we have examples where both the IF ... and the ELSE ... are executed making for a real mystery.

Although I will say more about PROC MODEL,SYSLIN, and SYNLIN in the 'what could be' section, a very serious weakness is that one can not control the output well enough to debug. If a system does not converge (or encounters a bad value-negative argument to the LOG() function while iterating), then you are forced to ITPRINT all of the iterations, even if you know from a previous run that iteration 20 and 21 will be enough. This produces a large paper cost. If the procedure had been designed by a practicing econometrician, such a weakness could not have survived.

Now there are many estimation techniques that are not available SAS but are used in econometric research. Among them are maximum likelihood techniques, BOX-COX regressions, a revised TSCREG and LAV supported by SAS, TOBIT regressions, Durbin's h-statistic, Householder transformations for exact DW statistics, but the most noticeable lack is the absence of any polynomial distributed lag estimation. I am not a proponent of Almon pol's at all, but they are rampant in empirical econometric literature, and if you are going to do empirical macro economics, like it or not, you will end up with needing Almon pol's. This is a serious weakness that keeps me embracing PEC and TSP far too often. One needs to have at least this technique available, and in SYSREG,REG and AUTOREG I at least these troubles of system estimation in AUTOREG recur. Other weaknesses are apparent in other procedures. ARIMA is late in developing transfer function analysis, but at least it has it now. But ARIMA still seems limiting due to the limitations of what values of the parameters it can handle (especially only allowing a maximum of second degree differencing and only up to 24 lags while the current literature seems to have third and fourth degree differencing many times and 32 lags). The inability to be able to statically simulate on a sample different from the estimation sample is also severe, especially in an ARIMA model where dynamic simulations tend to blow up more often than not.

Apart from procedures, the econometrician is sometimes faced with problems due to the organization of the DATA step. Any user migrating from TSP or TROLL who attempts to use the LAG function only goes away from SAS believing SAS is only for biometricians. Let me give you an example. In trying to overcome the deficiencies in AUTOREG, I was trying to construct weighted first differenced variables with different values of the weights. With such variables, one can use PROC REG to estimate an autoregressive equation by the Hildreth-Lu technique (albeit inefficiently). Consider the following program:

```
DATA TEST; DO OBNUM=1 TO 5; X=OBNUM;
   XLAG=LAG(X);
   END;
DATA TEST; DO 1=1 TO 4;
   RHO1=1; RHO2=1; RHO3=0; RHO4=5;
   OUTPUT;
   RHO1=1; RHO2=2; RHO3=0; RHO4=5;
   BADDIFFX=X-LAG(X)*RHO1;
   GOODDIFFX=X-XLAG*RHO1;
   OUTPUT; END;
```

Now from the program, since LAG is always executed the caveat about the use of LAG with an IF statement does not apply. But it is always executed, and hence does not pick up the lag value from the data set in the SET statement but rather the last execution. The variable BADDIFF does not get the desired result while GOODDIFF does. Statements such as

```
Y=LAG(Y) + .2*LAG(X) + .5*LAG2(X)
```

also cause problems depending upon whether y is in the data set or being created, but rarely does one get the desired result. One solution
is to have a \texttt{LAGDS()} function which gets its value only from a data set. The other solution is to warn everyone that the \texttt{LAG()} function is dangerous at best, and to never use it with any program control statements.

Well, now to What Could Be. First, SAS/ETS must have estimation techniques that are current in the literature. Anything else is a cop out of what SAS/ETS should be. If there can be video tapes for SAS training then there can be state of the art econometric techniques. Second, one should be able to combine estimation techniques—e.g., a system of equations, each equation having its own error structure, its own PDL structure, and requested diagnostics. Separate procedures will only work when they can be legitimately combined. PROC PRINCOMP is fine since it can be used with any procedure but as noted above PROC AUTOREG is not. Third, the documentation must be improved. A single page of TSP documentation tells more about autoregressive estimation, and the different techniques to handle it, than the whole of AUTOREG documentation. Fourth, the empirical econometrician's influence must be felt.

What I would like to do now is to describe what PROC MODEL, and SYSNLIN-SIMNLIN might be like in the future, with an eye on estimating and simulating large econometric models. Of course, if the techniques do well for large models, then they should do well for small ones, but as the number of equations and variables grows the difficulty of specifying and debugging an econometric model grows exponentially. For this reason, the model builder has a need for a specification scheme that is self documenting and flexible enough to allow for the kinds of debugging procedures that are commonly in use. PROC MODEL partially satisfies the documentation requirement in that one writes the equations in an algebraic form which is easily presented and understood. Parameters are specified symbolically and the researcher is left with the choice of how to differentiate variables from parameters. The specification of the system of equations in PROC MODEL becomes the "equation book" of the system.

But one should also be able to label equations as one can do in SYSTEM. Each equation should have its own EXOGENOUS, ENDOGENOUS, and PARAMETERS statements so that systems of equations can be pulled together or apart at the researchers whim. Estimation of each equation in such a system may be done separately from the system and rather than rely upon a good text editor (the full screen CMS editor is a candidate), equations should be labeled and referenced by name. Possibly we need a new PROC, for lack of a better name, PROC SPECIFY. Equations could then be coded there, and options given for printing them out both symbolically and with the parameters of each equation replaced with specific values.

For example,

\begin{verbatim}
PROC SPECIFY;
\end{verbatim}

\begin{verbatim}
CONS403 = A0 + A1*INCOME;
PARMS A0 300 A1 .75;
ENDOGENOUS CONS;
\end{verbatim}

In PROC MODEL then, the particular equations would be specified for use in SYSNLIN and/or SIMNLIN by name rather than direct coding. In fact the printing of the equations (with symbolic or value parameters) should also be possible so that the researcher could specify something like

\begin{verbatim}
PROC MODEL EQPRINT VPARMS;
EQUATIONS CONS INVEST GNP;
MODELS MONEY;
\end{verbatim}

The options EQPRINT and VPARMS would print the equations as specified in PROC SPECIFY with values replacing symbolic parameters. From the example above, we would have

\begin{verbatim}
CONS: Y = 300 + .75 \times INCOME
\end{verbatim}

printed. This structure allows a set of equations to be stored in a data set, and one can retrieve any set of equations that he desires for estimation and simulation. The "equation book" becomes self documenting just as DATA steps are self documenting in SAS. In both SPECIFY and MODEL there would also be a MODELS statement which would group equations together and the group could be referenced by name. Many different MODELS statements might be coded in SPECIFY to allow reference to slightly or substantially different lists of equations. In the MODELS statement, there would also be DROP and ADD statements so that equations could be dropped or added to the list of equations that a MODELS label references, e.g.

\begin{verbatim}
MODELS MONEY1 / DROP=(PHILLIPS);
EQUATIONS MYPHILLIPS;
\end{verbatim}

In the specification of each equation, the error structure and at least PDL's would be specified so that one might have an equation such as

\begin{verbatim}
CONS30: CONS = A0 + A1*INCOME(PDLINC,LAGINC) /
NLAG=CNLAG DW LAGS=( RHO );
ENDOGENOUS CONS;
EXOGENOUS INCOME;
PARMS A0 A1;
EQP ARMS PDLINC 3
LAGINC 24 CNLAG 4 RHO-;
\end{verbatim}

This specification indicates that there is a PDL on income in the equation and that the parameters of the PDL are to be set at (3,24) for the "equation book", the errors have a fourth degree autoregressive structure and the autoregressive parameters to be estimated are to be called \texttt{RHO1}, \texttt{RHO2}, \texttt{RHO3} and \texttt{RHO4}. The capability to be able to specify something like this will save a great deal of time for the user since he will be able to call the equation by name, use the default values for the option, or specify his own. This latter capability is
necessary unless every one has access to something like the CHS full screen editor and could retrieve the equations from the "equation book" for editing.

With a full screen editor, the ability to specify parameters of estimation with the equation would not be necessary—there would be PROC FSSPECIFY which would act like FSEDIT except that the data set would be the specification of the equations. With FSSPECIFY one would access equations, or models (list of equations) for full screen editing. This would be a super world in which some of us might not live, and hence I prefer to dream of the ability to specify equation estimation (and simulation) parameters symbolically.

In the estimation of the system, care must be taken in the specification of the instrument list, and what exogenous lags it should include. There should be an option in SYSNLIN to include lags without having to explicitly reference variables that are the lags themselves, and this option should be available for equations, groups of equations or the entire model (at least the latter).

The gracefulness of PROC SPECIFY and reformulated MODEL can be seen by attempting some debugging. To simulate a single equation of a large system, one simply codes:

```
PROC MODEL;
EQUATION CONS;
PROC SIMNLIN;
```
to do the job. Although this can be done now, one has to edit the system down to one equation in MODEL, and respecify the list of EXOGENOUS, EQUATIONS, etc., which is clumsy at best.

A new feature in SYSNLIN and SIMNLIN would be a "tag" feature which would automatically create new variable names for predicted, residual or simulated values for all endogenous variables in the system. Although TYPE serves this function in some uses, it does not facilitate comparison of predicted or simulated values from two or more "forecast" simulations, where exogenous or "add-factors" (controls, more or less) have changed. The problem without the tag feature is that comparing different simulations is harder. The TAG feature would allow one to create arrays such as ARRAY CONTROL C1-C323; and ARRAY POLICY P1-P323; which then can be used to compare the simulations easily. Of course one can now use the output data set from the control and policy simulations, concatenate them, do a PROC TRANSPOSE and then do the comparison with these as variable names. The TAG feature simply saves some of the users time in order to do the comparison he wants.

PROC SYSNLIN would have at least the Fletcher-Powell method if not other methods. Note that in the SPECIFY portion I have made specification of autoregressive errors as an option on the model statement, and PDL's as part of the specification. The documentation on SYSNLIN (pg. 15.8, ETS 1980) suggests formally specifying the equations describing the autoregressive structure as:

```
YTHAT = 80 + B1*Y;
Y = THAT + RHO*LAG(Y) - RHO*LAG(YTHAT);
```

The autocorrelation parameter specification in SPECIFY could simply produce the code displayed above, and might also try to save the first observation(s), for a true GLS estimation.

In order to debug nasty systems, the abilities of NLIN should be present—i.e.,
```
IF X<0 THEN DO;
FILE PRINT; N ITER X Y Z ...;
FILE LOG; END;
```
Forecasting requires additional capabilities that are not generally needed in insample simulations. These are related to specification of values of exogenous, and control variables or add-factors. A variable mentioned on a CONTROL statement simply gets the value specified for every observation in the data set. Add-factors are generally one observation additions as a constant term offset to an equation. One method of initializing add-factors is to statically solve the model for the last in-sample period. Add-factors are then set equal to -1*error for that period (or sometimes an average of n period errors) creating a so-called "null error" solution of the model. These values provide initial offsets the first forecast period, so that the model starts the dynamic simulation in the "right" place. The ADDFACT statement would not reference variables but rather equations, and it would have options for initialization such as averaging the equations error over the last n periods of static simulation. I.e., one would specify

```
ADDFACT CONS 5 INVEST 4;
```
to get an average of the last 5 periods of static simulation error to adjust the equation CONS and the last 4 periods to adjust the equation INVEST for the first dynamic forecast period.

Many other features need to be incorporated into SIMLIN. Most of these can be done now but clumsily. For example to do dynamic multipliers, one can statically simulate with two data sets, the second of which has I added to the beginning period value of the variable for which the multiplier is desired. Then the two output data sets must be merged, variables renamed, and differenced. The resultant values are the dynamic multipliers. But it would be much simpler to specify MULTANAL EXPORTS 1; which would automatically do this. A POLANAL statement (policy analysis) would do much the same thing, except that multiple variables and values would be specified. For example,

```
PROC SIMNLIN DATA=MNTX START=10;
POLANAL EXPORTS 10 20 10
GOVT 10 20 30;
```

83
would first simulate the model with the data in MINE, then simulate with EXPORTS and GOVT being changed by the values specified, simulate again and compare the results.

The reporting of results could be greatly enhanced with PROC MODEL EQPRINT; Currently, SYSNLIN stores its parameter estimates as part of the model data set. One should be able to print the model then with these values substituted for symbolic parameters—possibly accomplished with PROC MODEL EQPRINT VPARMS; as above but after estimation. Of course, SYSNLIN would need an OUTMOD= parameter so that the original model with its parameter values was not overwritten. The coefficients should be accessible, as should the covariance structure for other uses as we have in SYSGARCH OUTTEST= parameter.

In fact, after estimation, EQPRINT should make the equation print with the standard error printed below the coefficient. Just as importantly one needs to be able to compare different simulations. Usually one has say 300 variables, 4 quarters and at least two simulations. The presentation is best done by listing variables down the page and periods across. Although one can use PROC TRANSPOSE on the two data sets containing the two simulations and merge them together, the difficulty of keeping variable names straight, etc. begs for a procedure SIMCOMPR which would have some nice features for comparing simulations. Directly (printed output) and for creating data sets that are passed on to PROC PLOT and PROC GPLOT—the latter to make it easier to have two series of different magnitudes plotted on the same axis. The same end could be accomplished with a fancier GPLOT or PLOT.

I feel that I could go on and on about What Could Be, one can even imagine that we get PROC FSIMNLIN which would be like FSCALC in the format of a spreadsheet display, but non-linear simulation right there before your eyes, all in full screen display, with the ability to do real interactive policy analysis. I think that this will occur in the next 5 years as software develops for micro-computers but for now I am content to ask for PROC SPECIFY, EQPRINT options and the like. I hope that you as users will start those cards and letters coming into SAS to demand just this kind of software.

SAS Institute replies: the problem reported by Dr. Parks in doing multiple lags in static mode in SIMNLIN was solved over a year ago for the SAS82 release. However, SAS82 is not yet the production release. References to documentation are to the SAS/ETS User's Guide, 1980 Edition, which has now been superseded by SAS/ETS User's Guide, 1982 Edition.