ABSTRACT

The purpose of this paper is to demonstrate the use of the SAS/IML® software in a theoretical econometrics project that involves Monte Carlo experiments and non-linear search algorithms. We first impose decision rules in a principal components setting to compare the outcome of a pre-test estimator suggested by Mundlak (1981) against a more traditional mean square error reduction rule. The decision rules instruct the SAS/IML® software program to iteratively delete components one at a time until an optimum measure has been satisfied. Next we investigate the feasibility of a non-linear algorithm that iteratively searches for the optimum restriction to impose on a classical least squares estimator that will yield the same results as the variance reduction rule used in the Monte Carlo experiments.

INTRODUCTION

A number of authors have explored the optimality properties of principal components regression including Kendall (1957), Massy (1965), Marquardt (1970), Chen (1974), Greenberg (1975), Fomby, Hill and Johnson (1978), and Park (1981). Within the family of principal components estimation rules Greenberg (1975) showed that deleting components corresponding to the smallest eigenvalues brought the greatest reduction in the trace of the variance-covariance matrix of the estimated regression coefficients. Fomby, Hill and Johnson (1978) extended this optimality property to the entire class of restricted least squares estimators in the context of considering all possible linear restrictions on the regression coefficients.

Consider the standard regression model

\[ y = X\beta + \varepsilon \]

where \( y \) is a \( T \times 1 \) vector of dependent variable values, \( X \) is a \( T \times K \) matrix of explanatory variable values, \( \beta \) is a \( K \times 1 \) vector of regression coefficients, and \( \varepsilon \) is a \( T \times 1 \) vector of disturbances with a common mean of zero and a common variance \( \sigma^2 \).

Define \( Z = XA \) where \( A \) is the matrix whose columns, \( a_j \), are the eigenvectors of the \( XX' \) matrix with the columns of \( A \) ordered to correspond to the ordered eigenvalues, \( \lambda_j \), where \( \lambda_1 > \lambda_2 > \ldots > \lambda_K \). The ordered eigenvalues, \( \lambda_j \), also serve as the diagonal elements in the \( K \times K \) matrix \( \Lambda \). By appropriately partitioning the \( Z \) and \( A \) matrices as \( Z = \begin{bmatrix} Z_1 \mid Z_2 \end{bmatrix} \) and \( A = \begin{bmatrix} A_1 \mid A_2 \end{bmatrix} \) the principal components regression estimator may be expressed as

\[ \hat{\beta} = A_1'(Z_1'Z_1)^{-1}Z_1'y \]

\[ = A_1'(Z_1'Z_1)^{-1}Z_1'y \]

\[ = A_1'(Z_1'Z_1)^{-1}Z_1'y \] defining \( A^* = A_1' \) where \( A_2 \) has been replaced with a matrix of zeros. Of course, since the principal components are orthogonal to one another, it does not make any difference whether the classical least squares regression is run on the reduced set of principal components, \( Z_1 \), or on the full set of principal components, \( Z \), with the extra results zeroed out afterwards using the \( A^* \) matrix. These results are equivalent to applying least squares to the model

\[ y = X\beta + \varepsilon = ZA\beta + \varepsilon \]

to minimize

\[ \varepsilon'\varepsilon = y'y - 2\beta'AZy + \beta'A\Lambda\beta \]

subject to the restriction

\[ (A' \Lambda A)^{-1}b = 0. \]

The singular value and spectral value decompositions for the variance-covariance matrix of the classical least squares estimated regression coefficients are given by

\[ \text{Cov}(\hat{\beta}) = \sigma^2 (X'X)^{-1} \]

\[ = \sigma^2 A\Lambda^{-1}A' \]

\[ = \sigma^2 \sum_{j=1}^{K} \frac{1}{\lambda_j} a_ja_j' \]

In terms of the partitioned \( \Lambda \) matrix the classical least squares covariance matrix may be expressed as

\[ \text{Cov}(\hat{\beta}) = \sigma^2 A_1\Lambda_1^{-1}A_1' + \sigma^2 A_2\Lambda_2^{-1}A_2' \]

where \( \Lambda_1 \) and \( \Lambda_2 \) are defined in terms of the corresponding partitioning of the \( \Lambda \) matrix

\[ \Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \]

In contrast the principal components covariance matrix can be written as

\[ \text{Cov}(\hat{\beta}) = \sigma^2 A^*\Lambda_1^{-1}A_1^* \]

COMPUTER METHODS

The principal components regression model is intrinsically nonlinear so precise analytic solutions to the problem of component selection are not attainable by calculus techniques. Fortunately, state-of-the-art computers permit utilizing intensive Monte Carlo experiments and various search algorithms to examine regions of the parameter space. This section will first discuss Monte Carlo experiments that have been conducted over the past two years and then turn to search algorithms like Gauss-Newton.

Monte Carlo Experiments

Even though the true characteristics of a population cannot be known in practice, we assume they are known in the simulations. If the true population parameters, the \( \beta \) vector, were known, then how would the principal components estimator perform? That is, if the optimum estimator could be specified what would the results be in terms of mean square error? Partitioning our simulations after Hill (1987) we sampled 400 times and reported the average minimum mean square error for each of the 400 samples for least squares and principal components. The standard deviation, \( \sigma \), was allowed to vary so that performance over a range of the parameter space could be evaluated.
The following is the SAS/IML® software program that was used in the Monte Carlo experiments. The data set ONE contains the independent variable observations that comprise the matrix X and the dependent variable observations that comprise the vector Y. The estimated population parameters POPB are assumed to be the true population parameters and their classical least squares estimates are made utilizing a new vector BETA whose values are arbitrarily chosen to be close to POPB. The value of SEED is generated consistent with Hill (1987) to be a J vector with T rows where T = 36 in this case. EXPERNUM and SIGMA can be changed as needed.

```sas
PROC IML;
USE ONE;
READ ALL INTO WORKDATA;
X = WORKDATA[136,1:100];
Y = WORKDATA[136,12];
XTX = X' * X;
T = NROW(Y);
K = NCOL(X);
SEED = J(T, l, 1613228064);
SIGMA = 1.0;
BETA = {.1300, .4200, .2800, -.0100, .5200, .1900, .2100, .3900, .0800, .0100};
POPB = INV(X' * X) * X' * Y;
EXPERNUM = 400;
END;
```

The following set of statements describe the actual Monte Carlo run where a DO loop is iteratively executed as combined with the X matrix and BETA to produce a vector Y of random dependent variable observations. Thus, the new disturbances E, created using RANNOR, SEED, and SIGMA, is the basis for the calculation of the mean square error MSEHAT. PCLOOP is executed next, a routine that will delete components one at a time until K components have been deleted. The principal components pari of the program operates by deleting components one at a time until K components have been deleted and then calculating the mean square error as each component was selected deleted. It was difficult to make this operative because once a component was selected its corresponding eigenvalue was set to zero and on the next iteration that was then the smallest value. So, it was easiest to work with the reciprocals of the eigenvalues, the vector LL, and select the reciprocals from largest to smallest. Initially, LKEPT was equated to LL so DELETE will contain a zero in the location of the maximum value of LL and ones elsewhere.

Since UKIL is initially a vector of ones, when compared to DELETE it will then have a zero in the location of the selected value of LL and ones elsewhere. LL is then recalculated as LL = UKIL # LL so that the last chosen value has been deleted and LKEPT now has one value deleted. ADIM is created by diagonalizing UKIL and post-multiplying the eigenvector matrix A by this new matrix. The last column of ADIM that corresponds to the smallest eigenvalue now is all zeroes and the other columns are intact. The next iteration will select the next smallest eigenvalue and delete it in LKEPT. UKIL will then have two values set to zero so that the transformation will yield a matrix with two columns of zeroes that can act on the A matrix. When all eigenvector columns have been deleted the DO loop ends and PCMIN chooses the minimum mean square error for printing. Appendix A contains examples of LSELECT, LDELETE, LKL, LKEPT, LL, and APCat two different iterations.

START PCLOOP;
COUNT = 0;
ADIM = A;
DO UNTIL (COUNT = K);
BTLDA = A' * (ADIM * ZTPZI) * Z' * Y;
MSETIL = (BTLDA - BETA)' * X' * (BTLDA - BETA);
IF COUNT = 0 THEN MSETIL = MSEHAT;
ELSE MSETILV = MSETILV // MSETIL;
LSELECT = (MAX(LILV)) * ONEVEC;
LDELETE = LSELECT * LKEPT;
LKIL = (LKIL # LDELETE);
LL = UKIL # LL;
LKEPT = LL;
ADIM = A * DIAG(LKL);
COUNT = COUNT + 1;
END;
PCMIN = MIN(MSETILV) || MSEHATV(<.1:1.1); FINISH;
```

There is an abundance of criteria for guiding the researcher on selecting the correct model for a particular application; among the most popular are Mallows' Cp, Akaike's AIC, and Schwarz' BIC. George and Foster (1990) suggest a newer one based upon a calculation of risk inflation. Any number of stochastic stopping rules could be devised. One such rule could be modeled after the principal components stopping rule that says delete all those components that have statistically insignificant t-statistics in the regression run on all of the principal components. A more sophisticated variation of this is Mundlak's F-Testing for nonsignificant sets of principal components (see Mundlak, 1981).

Mundlak's rule uses the F statistic
\[
F = \frac{1}{D} \sum_{i=1}^{2} \frac{\lambda_i}{\lambda_{i+1}}
\]
where D is the number of components deleted and where
\[
2 \lambda_i \leq 1 \left( \frac{\lambda_{i+1}}{\lambda_i} \right)^{1-\lambda_i} \sum_{j=i+1}^{\lambda_{i+1}} \frac{\lambda_j}{\lambda_i} \left( \frac{\lambda_{i+1}}{\lambda_j} \right)^{1-\lambda_j}
\]
with \lambda_i representing the eigenvalues. Under this Mundlak rule, the optimal value of F_m (and, therefore, the optimal set of m eigenvector values to zero out) is such that either:

1. \( F_m < F_{n-R} \text{ table value} \) and \( F_{m+1} > F_{n-R} \text{ table value} \)
2. \( F_1 > F_{n-K} \text{ table value} \) and the matrix X has rank K.
3. \( F_{K-1} < F_{n-k} \text{ table value} \) and the matrix X has rank 1.
The pre-test estimator suggested by Mundlak performs in the same manner as the principal components method described above except that the decision rule calculates the number of components to delete and then uses the gully-washer approach to zero them out at one time rather than eroding them iteratively as was demonstrated above. Appendix B contains some of the most significant matrices that demonstrate the way the algorithm functions. The decision on when to stop deleting components is controlled by the value of PC_PVAL; the level of significance is chosen at .05 and the technique iterates until the component just deleted causes the p-value to fall below .05. The rule then says that the last component is added back and the routine is ended. In this case, the sixth one crosses the threshold so five components are deleted.

Specifically, K1 and K2 are used as counters at various stages of the routine and the matrices RANKT, NEWT, and, ORDERT merely determine the ranking of the t-statistics for deletion. SELECT chooses the appropriate element as K2 increases at each iteration so that the F-statistic can be calculated and then the corresponding p-value as given by PF. PC_PVAL accumulates the p-values for printing and possible later use in other statistics. Once PF falls below .05 it is set to 0 to insure the algorithm is halted. Appendix B also contains the calculations of mean square error for least squares (MSEHAT) and for principal components (MSETILV) which is an important aspect of our research on biased estimation regression. Pre-test estimators perform poorly in terms of mean square error (Hill and Judge 1987) and this case illustrates that fact.

The first order Taylor series expansion of f(R) is

\[ f(R) = f(R_0) + f'(R_0) (R - R_0) + \epsilon \]

Define

\[ Y = f(R_0) + f'(R_0) (R - R_0) + \epsilon \]

so that

\[ Y^* = Y - f(R_0) \]

so that

\[ Y^* = f(R_0) (R - R_0) + \epsilon \]

or

\[ Y^* = f(R_0) R + \epsilon \]

Then, define

\[ Y^{**} = Y^* + f(R_0) R_0 \]

so that

\[ Y^{**} = f(R_0) R + \epsilon \]

The least squares estimate of R is

\[ \hat{R}_{i+1} = ((f'(R_i))^T f'(R_i))^{-1} f'(R_i)^T Y_i \]

or alternatively

\[ \hat{R}_{i+1} = R_0 + ((f'(R_i))^T f'(R_i))^{-1} f'(R_i)^T Y_i \]

Now, f(R) is straightforward in practice because solving for it requires using the calculus of matrices. If the matrices were vectors the solutions would be relatively simple, but when they are not simple vectors the dimensions of the normal derivatives explode on the order of p and q so we had to develop our own techniques. Thus,

\[ f(R) = \frac{\partial f}{\partial R} = \frac{\partial [(X' X)^{-1} R' [R (X' X)^{-1} R]^{-1} R A]}{\partial R} \]

\[ = [(X' X)^{-1} \otimes I] \lambda_k (R (X' X)^{-1} R)^{-1} R A \otimes [k] \]
be respecified in vector terms, squares solution of for the more efficient Marquardt method which takes the form as on an initial value like The additional term merely changes the step length of the adjustment between iterations. Lambda takes depending upon differences between iterations. 

\[
\text{vec}(R_j^{+1}) = ([t'(R_j)]' [t'(R_j)])^{-1} [t'(R_j)] vec(V_j')
\]

This is the correct form for programming the least squares solution of the Gauss-Newton algorithm and is the basis for the more efficient Marquardt method which takes the form 

\[
\text{vec}(R_j^{+1}) = ([t'(R_j)]' [t'(R_j)])^{-1} [t'(R_j)] vec(Y_j').
\]

The additional term merely changes the step length of the adjustment between iterations. Lambda takes depending upon differences between iterations. The programming is as follows:

```
PROC IMPL;
X = (1, 2, 3, 0, 1, 2, 1, 2);
XTX = X' * X;
XTXI = INV(XTX);
CALL EIGEN(L, A, XTX);
J = NCOL(XTX);
XTXIRT = XTXI' R';
V = SHAPE(I,J,JSO);
D1_P1 = (XTXI' R') @ IJ;
D2_P1 = (X' XR1 R') @ IJ;
D3_P1 = (R' XTXI) @ IJ;
D4_P1 = SHAPE(4,JSO);
D4_P2 = SHAPE(A,JSQ);DER4 = D4_P2 - D4_P2
DER1 = (D2_P1 - D2_P2 - D2_P3)
DER2 = (D3_P1 - D3_P2 - D3_P3) + DER4;
```

**GEOMETRY OF DELETING ELEMENTS**

The following discussion borrows extensively from Fomby, Hill and Johnson's Advanced Econometric Methods wherein they develop an intuitive description of principal components regression. This particular effort concentrates only on the impact of deletion. The following discussion borrows extensively from Fomby, Hill and Johnson's Advanced Econometric Methods wherein they develop an intuitive description of principal components regression. This particular effort concentrates only on the impact of deletion. The original linear statistical model with least squares estimator maps the T rows of the X matrix into a K-dimensional space. In the case where K=2, Figure 3 shows the relationship among the X variables as anyone linear combination of the X variables variances, eigenvalues and eigenvectors. By construction the first principal component, \(Z_1\), captures as much of the variation in the X variables as any one linear combination of the X variables.
could possibly do. Thus the variance of $Z_1$ which is its eigenvalue, $\lambda_1$, represents the data scatter along the $a_1$ axis which represents the direction defined by the first eigenvector. In turn, the second principal component, $Z_2$, captures as much of the remaining variation in the $X$'s as any second linear combination that is orthogonal to the first linear combination could possibly do. The variance of $Z_2$ is its eigenvalue, $\lambda_2$. The greater variation along the $a_1$ axis implies that the information provided by the data along the $a_2$ axis is less precise than that along the $a_1$ axis.

The implication is that $a_2$ represents an "extra" dimension that is unstable and provides very little information. Therefore, estimation using only $a_1$ is more stable and reliable and uses almost all of the information in the original data.

**CONCLUSION**

This paper has illustrated the use of the SAS/IML® software procedure PROC IML in monte carlo experiments to evaluate the relative performance of a pre-test estimator proposed by Mundlak in a theoretical setting where it is assumed that the true coefficient parameters are known. Any number of other decision rules can be programmed so that the iterative deletion procedure can be stopped at any time.

Further, a more sophisticated nonlinear search algorithm was shown that searches for the optimal restriction to place on the eigenvector matrix when the model selection problem is cast in the restricted least squares context.

The authors may be contacted as follows:

**REFERENCES**


SAS/IML is a registered trademark of SAS Institute Inc., Cary, NC, USA.