Annals of Economic and Social Measurement, 2/3, 1973

PROGRAMMING SOFTWARE NOTES

MICRODATA PROCESSING PACKAGE

BY ATES DAGLI, IVA MACLENNAN, HYMAN SANDERS AND FINIS WELCH*

This note describes a package containing two programs, one for forming cross-product matrices, the other for calculating regression coefficients. Crossproduct matrices are the only permissible input to the regression program. The objective is to lower computational costs for regression problems based on large numbers of observations. Special features of the regression package are options for pooling and partial pooling of subsamples.

The *Crossproducts* program is designed to construct moment matrices for as many partitionings of a main sample as the user desires. The objective is to eliminate repetitive readings of a data file. Observations from the basic file are read and variables are suitably transformed for regressions in a user supplied subroutine. In this routine the user also specifies the subsample or samples to be used. The partitioning of the basic file is not required to be mutually exclusive.

The program includes a dummy variable option which utilizes efficient computation of moments. Classes of dummy variables must be mutually exclusive but need not be exhaustive. In a control card the user supplies dimensions of dummy classes and for each observation the user subroutine gives the location of the dummy variable assigned unit value for each dummy class. (It is not necessary to compute a vector of zeroes for each class.)

Efficiency is achieved in constructing moments by suppressing unnecessary multiplication and computing locations where non-zero additions occur. Suppose a problem includes N classes of dummy variables with N_i variables in each, and K continuous variables. Standard moment algorithms involve $(M + K)(M + K + 1)/2$ additions *and* multiplications, where $M = \Sigma N_i$. This program requires $N(N + 1)/2 + NK$ additions along with $K(K + 1)/2$ multiplications and additions. The saving over a large number of observations when M is large compared to N is obvious. Also, storage requirements are minimized by storing separate moments in upper and lower triangles of dimensioned arrays, and by dimensioning arrays specifically for each program run. The crossproducts routine prints summary tabulations of means, variances, and numbers of observations for each subsample. The matrices are punched or written on a tape or disk for future reference.

The companion regression routine, *Regress*, calculates OLS regressions from cross-product matrices formed elsewhere (possibly but not necessarily by *Cross-products*). Like many other routines, it will tackle a number of "main" problems, calculating individual regressions in a subproblem loop. Variable names and subproblem specifications can be stored and shared among the main problems. Subproblems allow suppression of the intercept and deletion of any specified

* Support for development of the program described here was provided by a grant from the Office of Economic Opportunity to the National Bureau of Economic Research.

set of variables. Options for printing of correlation and variance-covariance matrices are provided. *Regress* also contains a table option which prints summary tables for groups of subproblems with the same dependent variable. All of this is fairly standard. The program contains three more unique options: (1) Pooling samples, i.e. adding moment matrices. (2) Constructing linear combinations of variables in the input matrices. (3) "Partial pooling" of samples.

The pooling option is equivalent to constraining equality of estimated coefficients in subsample parts and offers a simple basis for the "Chow" test. This option is especially useful if the original partitioning is too fine, i.e., if some subsamples contain too few observations.

The linear combination option can be used for hypothesis testing, and also allows pooling of dummy classes when the original specification resulted in too few observations in some cells. The pooling and linear combination options are not exclusive.

The partial pooling option is the program's main feature. Using two samples (which may have been filtered through options 1 and 2), it allows regression estimates that constrain coefficient equality among some but not all variables in the samples. The model is of the form:

$$Y_1 = X_1\beta_1 + Z_1\gamma_1 + u$$

$$Y_2 = X_2\beta_2 + Z_2\gamma_2 + u.$$

Partial pooling constrains $\hat{\beta}_1 = \hat{\beta}_2$ while $\gamma_1 \neq \gamma_2$. For each set of two samples, the user designates a common set of variables and two sets of unique variables, one for each sample. Variables in common are matched between the samples so that when a variable from this group is used in a subproblem, coefficients on the matched variables are equated. When one of the variables from either of the unique sets is added, its calculated coefficient refers only to the sample from which it is drawn. The designations of unique and common sets are not mutually exclusive in the main problem, so between-sample equality can be changed from subproblem to subproblem. Furthermore, for any given subproblem it is not necessary either that any unique variables be included or that any common variables be included. This permits the Chow test to be based on two subproblems, one using no unique variables, the other using only unique variables. Notice also that when a variable is included in common and in the unique set of one of the samples, the calculated coefficient on the unique part is an estimate of the coefficient difference between the samples; the T statistic which is printed gives a simple basis for testing the hypothesis of between-sample coefficient equality. This partial pooling option, along with the facility for constructing linear combinations of variables, allows a wide array of linear hypotheses to be tested. Because the regression program aggregates samples and/or variables but cannot disaggregate, the incentive for fairly fine stratification in constructing moments is obvious.

These programs are by no means in their final forms, and are hence continually subject to revision. However, they have been thoroughly tested and debugged. They are available at the NBER and Columbia University via WYLBUR, and require about 200K for execution.

Several cautionary notes are addressed to potential users. The microdata package as presently constituted is written in the FORTRAN IV language and

executes off an IBM FORTRAN IV Level G compiler under OS. Both programs utilize options peculiar to IBM Fortran and would require minor modifications to run with non-IBM software.

Care should be taken in the generation of crossproduct matrices because too fine a classification within dummy classes may result in a null row. Similarly, a large range of values for a given variable will adversely affect the degree of precision. To protect against the latter occurrence as much as possible, data are collected in counters which are double-precisioned.

A final note concerns the degree of accuracy attained upon inversion of the crossproducts matrix. Whenever wide ranges of values occur, as happens in the case of dummy and continuous variables grouped together, precision will be lost. The degree to which this will be so is somewhat dependent on the inversion algorithm utilized. *Regress* employs the Gauss–Jordan method (see any numerical methods text for a description), which makes the best of a bad situation.

A detailed program description may be obtained from Helen Smith, NBER, 261 Madison Avenue, New York City 10016.

National Bureau of Economic Research

received: December 16, 1972
revised: April 9, 1973