On the advantage of using two or more econometric software systems to solve the same problem

Houston H. Stokes
Department of Economics (MC 144) College of Business Administration, University of Illinois at Chicago, 601 South Morgan Street, Room 2103, Chicago, IL 60607-7121, USA
Tel.: +1 312 996 0971; Fax: +1 312 996 3344; E-mail: hhstokes@uic.edu

This paper illustrates the dangers of relying on just one software system to solve what appears to be a fairly routine probit model. It is shown that quite different results can be obtained using various well known software systems. In this example, at a superficial level of analysis, the differences were traced to the default convergence tolerances that are built into the statistical software. At a deeper level of analysis, the effect of the convergence tolerance on the solution indicates that the maximum likelihood estimate does not exist due to a formerly undetected quasi-complete separation problem. Using a Multivariate Adaptive Regression Splines (MARS) model to relax the assumption of constant coefficients, further implications are drawn from this dataset.

1. Introduction

McCullough and Vinod [18] surveyed the growing literature on the problems of statistical software reliability. In their view, user errors can be caused by poorly designed code, outright errors in the coding or due to misuse of the package. While such problems can and do occur in linear models, as was shown in classic work by Longley [14,15] and others, in nonlinear models the problems can be substantially more acute. With the greater inherent complexity of a nonlinear model comes greater potential for problems. In a recent paper on nonlinear estimation, McCullough and Renfro [17] stress two major areas of concern. First, for many nonlinear models one algorithm might produce an answer while others may fail. Second, even if both algorithms produce answers, one may be more accurate. Their paper traces estimation accuracy difficulties to differences in step length, convergence criterion and the way that derivatives are calculated. The present paper provides an example that documents a case where there was an undetected underlying problem in a nonlinear probit/logit model that had been widely published in various places for many years. The problems of this model were detected by accident during a routine replication exercise. After careful analysis and experimentation, it was found that only by varying the convergence tolerance was it possible to determine that in fact there was no maximum likelihood solution to the model, which turned out to be a undetected quasi-complete separation problem.

The paper starts with a brief discussion of the nature of the problem and how it was found. Next, the symptoms of the problem are investigated. To make sure this

Table 1
Variables in the McManus capital punishment dataset

Variable	Description
T	Median time served convicted murders
Y	Median family income of families in 1949 (thousands of dollars)
LF	Labor force participation rate in 1950 (expressed as a percent)
NW	Proportion of population that is nonwhite in 1950
D2	Dummy Variable, 1 for southern states, 0 for others
PX	Average number of executions 1946–1950/convictions in 1950
D1	Dummy Variable, 1 $PX > 0$, 0 otherwise

Source: Maddala [16]. Original data from McManus [20].

difficulty was not related to one or two particular software packages, a number of well known software systems are tested and their results reported. Since software systems are "black boxes" in that the calculations are hidden in compiled code, the problem was attempted with the IMSL routine DBCONF, which is available in B34S, and FMINUNC, which is available in Matlab. Here the calculation is completely exposed. To facilitate replication on the part of readers, all input control files, data and output and log files are available on-line to facilitate further experimentation and study. For the DBCONF and Matlab results the exact setups are shown in Tables 2 and 3. An important goal is to discuss the symptoms of this problem to alert researchers when their software is not able to detect the exact problem. Finally, a Multivariate Adaptive Regression Splines (MARS) model is used to relax the assumption of constant coefficients.

2. Data and preliminary results

Strange results were obtained when the author used what was thought to be a stable probit test case reported by Maddala [16] to test the SAS PROBIT command. Maddala, then at the University of Florida, had estimated a probit model using data on the deterrent effect of capital punishment that had been reported and analyzed with baysian methods by McManus [20] in 1985 . The dataset consisted of a cross section of 44 states and contained data on capital punishment. Key variables and their descriptions are listed in Table 1.

Maddala [16] in 1988 and later in 1992 reported an OLS model

$$\begin{array}{ccccccc} \hat{D}1 = 1.993 & + 0.00146T & + 0.658Y & - 0.055LF & + 1.988NW & + 0.343D2 & \\ (1.33) & (0.001) & (0.240) & (0.028) & (0.759) & (0.189) & (1) \end{array}$$

where standard errors are reported under the coefficients.[1] Maddala next estimated the above model using probit (his results are reported in Table 2 as column 6) and

[1] Maddala actually reported t scores. The SE's were calculated from the t's. Variable names are defined in Table 1.

Table 2
Probit results for extended model

	B34S	SAS	RATS	LIMDEP	Stata	Maddala	B34S	B34S	B34S
Column	1	2	3	4	5	6	7	8	9
CON	6.915406	−6.9155	6.9155	6.9155	6.915513	6.92	6.915403	6.915403	6.915403
	11.32239	11.3225	11.3225	11.32251	11.3225	11.344	11.32242	11.3224	11.3224
T	0.011312	−0.0113	0.0113	0.011312	0.011312	0.0113	0.011312	0.011312	0.011312
	0.005656	0.0057	0.005656	0.005656	0.005656	0.00565	0.005656	0.005656	0.005656
Y	6.461074	−6.4611	6.4611	6.46112	6.4611	6.46	6.46107	6.46107	6.461074
	3.14832	3.1484	3.1484	3.1484	3.1483	3.1512	3.14833	3.14833	3.148326
LF	−0.40928	0.4093	−0.4093	−0.40929	−0.40929	−0.409	−0.40928	−0.40928	−0.40928
	0.257213	0.2572	0.2572	0.2572	0.2572	0.2572	0.257213	0.257213	0.257213
NW	42.49808	−42.4983	42.4983	42.49827	42.4983	42.5	42.49808	42.49808	42.49808
	20.74961	20.7498	20.7498	20.7498	20.7497	20.732	20.7497	20.7497	20.7497
D2	4.39004	−6.8494	8.0863	7.197776	*	4.63	5.256471	6.162708	7.09329
	46.3237	53,170.4	4,647,813	114,550.1	*	115.75	325.662	3,802.26	73,594.5
LogL	−8.64041	−9.01665	−8.64038	−8.64038	−8.64038	na	−8.64039	−8.64039	−8.64039
Tol	1.00E-05	0.001	0.00001	0.0001	1.00E-6	na	1.00E-07	1.00E-09	1.00E-12
Iterations	11.00	na	36.00	29.00	6.00	na	15.00	20.00	24.00
Cond	6.46E+08	na	na	na	na	na	3.86E+10	5.29E+12	1.98E+15

For data sources see Table 1 and text. LogL is the Log Likelihood. Tol = the convergence tolerance used. Tol set to default for all but columns 7–9. Cond is the condition of the variance covariance matrix.

remarked "surprisingly, D2 is not significant but all other coefficients have the same signs as in the linear probability model." Stokes [24] used this model to illustrate probit models using the B34S(R) software in 1991, and later in 1997 did further analysis using the Friedman [9] MARS model. The B34S results for Maddala's Eq. (1) estimated as a probit model are listed in column 1 in Table 2. Here the log likelihood function is reported and the condition of the variance covariance matrix is given as 6.46E + 08. The difference between column 1 and 6 is in the D2 coefficient, which was 4.39 for B34S and 4.63 for Maddala. Initially, the importance of this was thought to be a typesetting error, since in addition to the equation in column 6, Maddala reported a subset probit model that removed D2.

$$\begin{aligned}\hat{D}1 &= 10.27 + 0.0094T + 5.55Y - 0.437LF + 50.25NW\\ &\quad\ (10.48)\ \ (0.0051)\quad (2.817)\quad (0.257)\qquad (20.1)\end{aligned} \tag{2}$$

which was exactly replicated with B34S for all coefficients and their standard errors.

At a later time in order to benchmark the SAS PROBIT command, Maddala's Eq. (1) was run and is reported in Table 2 as column 2.[2] In comparison with the

[2]Due to software design, SAS calculates the cumulative normal distribution of −XB rather than XB,

B34S result listed in column 1, we note the Log Likelihood function is virtually the same and all coefficients except D2 are the same except for a sign change. For D2 SAS had reported -6.8494 with a suspect SE of 53,170.3. Subsequent testing with RATS, reported in column 3, and LIMDEP, reported in column 4, found the same pattern. Again, the Log Likelihood agreed and all coefficients were the same, except D2. For the RATS Linux version 5.01 PRB command, the SE became 4,647,812.5029 while for LIMDEP 7.0 on Linux (compiled with the Lahey LF95 compiler from Fortran), the SE was 114,550.1. Point estimates of the D2 coefficient were 8.0863 and 7.19776, respectively. As a final test of packages, Stata version 7.0 for Linux was tried. Stata reported "note: d2 $\sim= 0$ predicts success perfectly. d2 dropped and 15 obs not used." The Stata results are shown in Table 2, column 5. The reported coefficients and log likelihood (-8.64038) for this "reduced model" were found to be nearly the same as was found by the other systems for the complete period including D2. What Stata appeared to have done is to just not report the coefficients for D2. The cryptic Stata message might cause some confusion to the average researcher since what Stata reports is *not* Maddala's model (2), which is a probit model estimated for all 44 observations without D2 on the right-hand side. If the Maddala Eq. (2) is run, all programs replicate the results reported by Maddala. In the next section a number of tests are run to attempt to understand the nature of the problem that is causing the different results.

3. A discussion of the symptoms of this problem

The B34S probit command has a tolerance that defaults to 1.0E-5.[3] It was found that if this is reduced to 1.0E-7 (see results reported in column 7 in Table 2), the D2 coefficient increases to 5.256471 and the SE increases to 325.662. Further reductions to 1.00E-9 and 1.00E-12 cause the coefficient of D2 to increase to 6.162708 and

which results in all coefficients having the opposite sign. This is a potential problem for users of this software who do not carefully read just what is being estimated and/or fail to validate the SAS results with other software.

[3]The B34S PROBIT command was originally obtained from Mathematica Policy Research in the middle 1970's. This tolerance is a bit large in comparison with some other software systems. The code has been substantially altered by the developer of B34S, although the basic structure of the program has remained the same. The program is 100% double precision and older inverters have been replaced by the LINPACK [7] Cholesky routines DPOCO/DPOFA/DPODI, which are used to invert and calculate an estimate of 1 over the condition of the variance covariance matrix. This has been converted to the condition. It should be noted that this estimate differs slightly from the estimate provided by LAPACK [3]. SAS is documented in Barr, Goodnight, Sall and Helwig [4], while RATS is documented in Doan [8]. Greene [11] documents LIMDEP, while Stata is documented in [25]. White [26] documents SHAZAM version 8 while EViews is documented in [23]. The versions of the software used were: B34S 8.67E, Rats 5.01, Limdep 7.0 and Stata 7.0. All initial runs were on a PC running Linux Red Hat 7.1. The same runs are made on W2K, slight differences will be observed for the extreme cases, most likely due to run- time library differences. Setups for this problem are distributed with the B34S.

Table 3
Implementation of the probit model in B34S

```
b34sexec matrix;
call loaddata;
mask1 = (d1.eq.1.); mask2 = (d1.eq.0.);

program test2;
xb=(cc+ t_coef*t + y_coef*y + lf_coef*lf
      + nw_coef*nw + d2_coef*d2);
func= mlsum((mask1*probnorm(xb))+(mask2*probnorm((-1.)*xb)));
return;
end;

rvec=array(6:  10., 0.01, 5., -0.4, 40., 3.);
ll=  array(6:-100.,-100.,-100.,-100., -100., -100.);
uu=  array(6:  100.  100.  100.  100.  100., 100.);
call echooff;
call cmaxf2(func :name test2
  :parms cc t_coef y_coef lf_coef nw_coef d2_coef
  :ivalue rvec :maxfun 20000 :maxg 20000
  :maxit 100000 :lower ll :upper uu :print);
b34srun;
```

7.09329 and the SE to rise to 3,802.26 and 73,594.5, respectively, (see columns 8 and 9). If the value of 1.00E-13 is used, the condition check of the hessian matrix fails and the B34S probit command stops because the hessian matrix cannot be inverted. Note that for the B34S equations (columns 1 and 7–9), the condition of the variance covariance matrix is reported. As the tolerance is reduced from the default in column 1 to 1.00E-12 in the equation reported in column 9, the condition increases from $6.46E + 08$ to $1.98E + 15$. The results of this first experiment suggest that the puzzle is related to the convergence tolerance.

To confirm that these findings were not unique to the B34S, the tolerance in RATS (CVCRIT) was raised from the default of 0.00001 to 0.01 and 0.1, respectively. This change in the opposite direction resulted in the estimated D2 coefficient falling to 7.8911 and 3.3928 and the standard error falling to 1,702,804 and a more reasonable 8.17, respectively. This experiment confirms the sensitivity of the D2 coefficient to the tolerance is not a unique feature of the B34S system. The importance of the experiments with tolerance is that when using default settings many software systems may appear to converge when in fact they have not.[4] If this happens, rank problems

[4]The other possibility is that the default tolerance in a software system is such that a rank message is obtained. The McKelvey-Zavoina [19] program MPROBIT, which is part of B34S, is just such a program. Here, unless the tolerance is increased from the default of 0.1E-5 to > 0.02, the hessian matrix is found to be not full rank. The Press-Nerlove [22] LOGLIN and the Kawasaki [13] MLOGLIN programs, also part of B34S, fail to estimate the problem using logit models when default settings are used. With the default number of iterations, LOGLIN did not converge. When the iterations were increased, a rank error occurred on the hessian matrix. Similar problems were obtained with MLOGLIN. The fact that all three systems fail with default settings protects the user but does not give an insight into the nature of the problem.

Table 4
Implementation of the probit model in MATLAB

```
function y=testcase(x,d1,xx);
% sets up a PROBIT using normcdf function
% x are the coef, d1 is the left hand side
% xx is the data vector

xb=xx*x';
dd=abs(d1-1.);

y=-1.*(sum(d1.*log(normcdf(xb)))+ ...
  sum(dd.*log(normcdf((-1.).*xb))));
guess=[10.  0.01 5.  -0.4 40.  3.  ];
options=optimset('MaxFunEvals', 2000,'MaxIter',2000)
xx=[ones(size(t)) t y lf nw d2];
[x,fval,e,o] =fminsearch('testcase',guess,options, d1,xx);
[x2,fval2,e,o,g,h]=fminunc('testcase',guess,options,d1,xx);

disp('Answers using fminsearch & fminunc for Base Case')
[x' x2']
disp('Log Likelihood Function found')
[fval' fval2']
disp('Gradint');
g
disp('Hessian');
h
ih=inv(h);
se=sqrt(diag(ih));
tstat=x2' ./se;
disp('Coef SE tstat')
[x2' se tstat]
```

Note: The data loading steps have not been shown in Tables 3 and 4.

on the hessian matrix predicted by the theory may not be seen. This problem is investigated first by using the IMSL [21] routine DBCONF, to obtain measurements of the gradient and Matlab [5] to verify the findings. The setups for these runs illustrate the probit calculation in two 4th generation programming systems and are given in Tables 3 and 4, respectively. A probit model takes the form $P(y_i = 1|X_i) = F(X_i\beta)$, where y is a 0–1 left-hand variable and X_i is a set of explanatory variables for the *i*th observation. $F(\ )$ is the standard Normal c.d.f. The probit model is estimated by maximizing $L = \sum_{Y_i \neq 0} \log(F(X_i\beta)) + \sum_{Y_i=0} \log(1 - F(X_i\beta))$ over β. In the B34S MATRIX command the function PROBNORM is $F()$ while the function MLSUM sums the logs. The mask1 and mask2 variables facilitate vectorization of the problem, since parse costs of the calculation are fixed no matter how large the sample size. The variable rvec sets an initial guess and uu and ll set upper and lower bounds. In the call to CMAXF2, the function name (func) and its program (test2) are passed as well as some switches.

The Matlab implementation requires a function (y) to be specified. Here d1 and dd are the mask variables and NORMCDF the cumulative normal distribution. After

Table 5
Experiments with IMSL routine DBCONF

	Constant	T	Y	LF	NW	D2	LogL	# It	Grad Tol
Setup 1									
Coef	6.9156	0.011312	6.461085	−0.40929	42.4984	4.6838	−8.64038	199	6.06E-06
SE	10.44	0.005358	2.94959	0.23754	17.5124	138.9			
Grad	1.22E-08	−7.86E-07	−2.17E-08	−1.69E-07	7.27E-09	−1.12E-05			
Setup 2									
Coef	6.9155	0.011312	6.461116	−0.40929	42.49827	5.397677	−8.64038	15,585	1.00E-13
SE	0.001132	8.92E-05	2.75E-04	2.09E-04	1.44E-02	3.31E+00			
Grad	2.44E-08	−1.40E-07	−8.69E-09	2.81E-08	6.61E-10	−3.28E-07			

setting some switches with OPTIMSET, the Matlab commands FMINSEARCH and FMINUNC are used to obtain the "solution." From the hessian matrix we calculate the SE and print the results.

As is clear from Tables 3 and 4, to use DBCONF under B34S or FMINUNC under Matlab requires that the user supply the model in a matrix language. Apart from this input, the rest of the calculation is done with DBCONF or FMINUNC. We next discuss the DBCONF results reported in Table 5. Setup 1 listed in Table 5 shows the solution of the model using the default settings for all input parameters. The SE is calculated as the square root of the diagonal of the hessian matrix. Of all the results reported so far, these are closest to Maddala's original findings and may give us a clue about the software settings he used. Inspection of the gradient[5] shows a value -1.12E-5 for the D2 coefficient, which is consistent with the "convergence" that was found with the B34S probit command. Setup 2 substantially lowers the gradient tolerance to 1.00E-13 and results in 15,585 iterations, no change in the Log Likehood and a coefficient and SE of D2 of 5.3976 and 3.305, respectively. The message "Scaled step tolerance satisfied. May be a local solution. Or progress too slow. Adjust STEPTL." suggests problems. The SE's are quite different and should not be taken seriously. The gradient for D2 decreased to -3.28E-07. The fact that changes in the tolerance change the D2 coefficient and its SE is a symptom that is consistent with D2 not being identified.[6] These findings again point up some of the

[5]The suggestion to investigate whether the gradient of the suspect parameter (D2) at the alleged maximum was not zero was made by a reviewer. Using the B34S probit command, the condition for the hessian matrix for equation 1 was 6.46E + 8 and not appreciatively different from the condition found when estimating the subset model in equation (2) that did not contain D2 where it was 1.22E + 8. However, when a smaller tolerance was used, the condition increased. Gould and Sribney [10], who are developers for Stata, discuss on page 119 situations in which as the iterations proceed the negative hessian matrix cannot be inverted and a direction cannot be calculated. They note that in such cases Stata will give a message that the likelihood is "not concave." During the solution of this problem no such message was observed from Stata.

[6]The MATLAB optimization routine FMINUNC [5] was used to verify what was found with DBCONF. Using the default settings the D2 coefficient was 4.1136 with SE 5.7790. If the TolX and TolFun are set to 0.10E-13, the coefficient rises to 15.833, the SE increases to 10,317 and we get a rank warning for inverting the hessian matrix. As the tolerance is lowered, the SE's move for all coefficients due to inversion problems.

pitfalls of running programs with their default values, a point made by McCullough and Renfro [17].[7]

Sorting the dataset with respect to D1, the dependent variable, reveals that when D1 $=0$, D2 is always 0 but then D1 $=1$, D2 can be 0 or 1. This means that if we find that D2 $=1$, we know for certain that D1 $=1$. There were 15 such cases. Once the sort was studied, the formerly cryptic message in Stata was understood. All 15 such cases were detected by Stata. What is strange is that if Stata really believed its own message, it would have either reported nothing (the best decision) or reported the subset model in Eq. (2). Just not reporting the D2 coefficient but leaving all other coefficients in the model is a strange and confusing choice. The obvious question was to see if other widely used software systems were able to detect this problem.

If Eq. (1) is estimated using EViews version 3.0, the log likelihood is found to be -8.64038, the model converges after 27 iterations and the D2 coefficient and its SE are 8.591597 and 360,990,038, respectively, suggesting problems to most experienced researchers. However, if SHAZAM version 8.0 is run, we get convergence after 8 iterations with the log likelihood -8.6428 and D2 coefficients and SE as 3.2825 and 7.0084, respectively. SHAZAM reports the tolerance was 0.001. Other coefficients are close to the values found for other systems but still differ. For example, the LF coefficient was -0.40930 with SE 0.25713, while most other software systems found -0.40928 with SE 0.2572. These findings suggest nonconvergence due to the large tolerance that would not be readily apparent to a researcher not using other software systems. TSP [12] Version 4.5 provides a better message to the user. When Eq. (1) is estimated, TSP reports "Quasicomplete Separation: some observations perfectly predicted for choice… D2 $>$ 0.00 perfectly predicts D1 $=1$. Delete D2 and all obs. which satisfy this. Estimation would not converge; coefficients of these variables would slowly diverge to $+/-$ infinity – the scale of the coefficients is not identified in this situation." TSP makes reference to Albert and Anderson [1] and Amemiya [2] for a further discussion of Quasicomplete Separation. In the next section this problem is discussed in more detail.

4. Diagnosis of the problem

The key reference to the problem of Complete Peparation and Quasicomplete Separation is Albert and Anderson [1], which is the basis of the reference in Amemiya [2]. No other econometric book seems to address this issue except for Davidson and

[7]The B34S control setup that runs all the tests, the B34S log and the B34S out file for the job are available on the web under the page www.uic.edu/ hhstokes , which points the user to the B34S ftp location. The file names are probit.b34, probit.log , probit.out, probit2.b34 and probit2.out. Complete data and output are given for Tables 2 and 5.

MacKinnon [6], whose book is in press and contains an excellent summary of the problem.[8] Define the probit log likelihood function as

$$l(y,\beta) = \sum_{i=1}^{n}(y_i \log(F(X_i\beta)) + (1-y_i)\log(1-F(X_i\beta))) \tag{3}$$

where y_i is the left-hand side variable assumed to be 0-1 and X_i is a vector containing the *i*th observations of right-hand side variables. Complete Separation or a Perfect Classifier Problem occurs if there is a linear combination of variables such that

$$y_t = 0 \text{ if } X_t\tilde{\beta} < 0 \text{ and } y_t = 1 \text{ if } X_t\tilde{\beta} > 0 \tag{4}$$

The symptom of this problem is that one or more of the parameters will increase without bound since the log likelihood in Eq. (3) approaches zero if $\beta = \gamma\,\tilde{\beta}$ and $\gamma \to \infty$. This pattern was not observed. Quasicomplete separation occurs if there is at least one observation with $y_i = 0$ and $X_i\tilde{\beta} = 0$ and at least one other observation with y_i and $X_i\tilde{\beta} = 0$. Recall that the sort showed that for all cases when D1 =0, D2 was 0 and when D1 = 0, D2 was 0 and when D1 was 1, D2 could take on the value 0 or 1. For the McManus data we can identify all coefficients except D2. There is no unique D2 coefficient, since we can set it to any value and for the case when D2 = 1 still predict with 100% certainty that D1 was not equal to 0. In this situation the likelihood function is not zero. Although it exists in theory, it can never be reached numerically, since as the D2 coefficient increases in value the density of the cumulative normal distribution approaches zero, which results in rank problems when inverting the hessian matrix. As is noted in Albert and Anderson [1], in such a situation the maximum likelihood estimator does not exist.

5. A strategy of prevention

The main lesson learned in this exercise is to not accept the output of one program without further testing with other software. The default convergence tolerances may give the illusion of convergence when in fact this has not occurred. If one software system is used, it is a good idea to vary the default nonlinear estimation assumptions to see the results as was suggested by McCullough and Renfro [17]. While it is always a good idea to check the gradients, in this case this did not flag the problem. The condition of the hessian matrix should also be checked on a routine basis. If the condition increases as the tolerance is lowered, this is cause for further concern and warrants further checking, since it suggests the hessian matrix may be moving toward singularity, which is a symptom of quasicomplete separation.

[8] Evelyn Lehrer, who worked with Press and Nerlove [22] on LOGLIN, characterized the difficulty as an "empty cell problem." In the context of the sort, this would mean that for the case where D1 = 0, D2 is never 1. The key chapter in Davidson and MacKinnon [6] is 11.

The dramatic increase in the SE as the tolerance is lowered is another giveaway that the hessian matrix is unbounded. It is clear that the estimated SE's should be inspected closely. The SAS SE of 53,170.39, the EViews SE of 360,990,038 and the RATS SE of 4,647,813 raise red flags to veteran empirical workers. At issue is both the parameterization of the model (if it is probit what variables are on the right?) and the more general question of whether in fact a hyperplane exists for this problem. To obtain some insight into the underlying model, in the next section we relax some of the assumptions of the probit model and investigate interaction terms using MARS.

6. Further results when some assumptions of the model are relaxed

The Multivariate Adaptive Regression Splines (MARS) technique was developed by Friedman [9] to relax the assumption of constant coefficients at all levels of the right-hand side variables. The MARS technique provides a spline-based method to search for interactions and threshold effects in the data that are implicitly assumed away in a more traditional probit model. Only brief discussion of the theory will occur here. More detail is provided in Stokes [24] and Friedman [9]. The first step in identifying a MARS model is to determine the maximum number of basis functions or knots and the maximum number of variables in each basis function. In this analysis the number of variables in each basis function is limited to a maximum of 2. If the maximum was set as 1, a simple spline model with no interactions is estimated.

Define $(x_i - \tau_i)_+ = x_i - \tau_i$ if $x_i > \tau_1$ and 0 otherwise. Assuming a linear model, if a MARS model is fit in a situation in which there is no break in the data, then the model found should be

$$y = \alpha + \sum_{i=1}^{k} \beta_i (x_i - \tau_i)_+ \tag{5}$$

where τ_i is the minimum of the x_i vector for $i = 1, \ldots, k$. The linear OLS model is a special case of the MARS model. If τ_i is not set at the minimum of the x_i vector, we have a knot or in words a change in the coefficient that is related to a particular value in x_i vector. The Friedman MARS software allows estimation of a logit form of this equation, not the probit form.[9] Since logit is similar to probit, a logit MARS model will be used in the next experiment. Define

$$f(X) = f(\alpha + \sum_{i=1}^{k} \beta_i (x_i - \tau_i)_+) \tag{6}$$

[9]While it would be possible to modify the Friedman [9] MARS library Fortran source to add probit, this was not the focus of this paper. In addition, the author did not want to introduce any possible additional sources of error arising from such modifications. The reason to attempt the problem with MARS logit was to investigate dropping the hyperplane assumption and allow modeling interaction points.

The logit MARS model becomes

$$F(Z_i) = (1/(1 + e^{-\hat{f}(X)})) \tag{7}$$

The MARS procedure calculates a modified generalized cross validation statistic (GCV) for the complete model and a GCV_i if the ith variable is removed.[10] If the ith variable was important, then $\mathrm{GCV}_i > \mathrm{GCV}$. If the ith variable did not add to the model, then $\mathrm{GCV}_i < \mathrm{GCV}$ or the penalty of adding that variable was not offset by greater predictive accuracy obtained by including that variable. Such a technique is useful in telling what variables are important in the model. The GCV_i vector can be normalized to form the normalized relative variable importance vector NRVI_i, where each variable is rated on a scale of from 0 to 100.

$$\mathrm{NRVI}_i = (100 * (\max(0, (\mathrm{GCP}_i - \mathrm{GCV})))/A0, \tag{8}$$

where

$$A0 = \max(\mathrm{GCP}_i - \mathrm{GCV}) \tag{9}$$

for $i = 1, \ldots, k$ variables in the model. In the stepwise forward-knot placement phase of model estimation, the GCV is shown as each basis function enters the model. If the basis function adds to the model, the GCV will fall. If the basis function does not add to the model, then GCV can rise. After backward stepwise elimination, the software produces a table that shows the basis functions remaining in the model and the increase in the GCV that would occur if the basis function was removed from the model. In OLS models without interaction terms, there is a one-to-one mapping of basis functions and variables. Once knots are introduced to the analysis, there is the possibility that one variable can enter into the model as part of one or more functions.

For the McManus data the MARS logit model estimated was as follows:

$$\begin{aligned} \mathrm{D1} &= 0.9951 - 0.1303(\mathrm{T} - 164.)_+(0.061 - \mathrm{NW})_+ \\ &\quad -0.2788(164. - \mathrm{T}_+)(0.061 - \mathrm{NW})_+ \\ \mathrm{GCV} &= 0.08529 \end{aligned} \tag{10}$$

The estimated MARS model was estimated where there was a maximum of 5 knots, and the highest order interaction was 2. Setting the number of knots larger had no effect on the estimated model. Recall that in equation (1) the probability of the state having capital punishment was positively associated with T (medium time served), Y (median family income) and NW (proportion of the population that is nonwhite). While the probit model had three significant variables, the MARS/logit model had only two, T and NW, with Y dropping out. Interestingly, the deterrent

[10]For details on the construction of this statistic and other MARS related questions, see Friedman [9].

effect of time served (T) only affected whether the state had capital punishment if the proportion of the nonwhite population was < 0.061. If this condition was met and if the median time served was > 164 months, an increase in T *lowered* the probability that the state had a death penalty (see coefficient -0.1303). However, if the median time served was < 164 months, then increases in T were positively associated with the state having the death penalty (-0.2788 multiplied by $-\mathrm{T}$ is positive). Note that the probit model, by construction, was not able to detect these level effects and just found the positive effect. What the MARS logit model suggests is that if the median time served for murders is below 164 months, the population views the death penalty as a complementary punishment with prison time. If the level of sentencing produces longer median sentences above 164 months, the death penalty is a substitute for a prison time.

To access the models estimated, the variance of the residuals was 0.065002 in the probit model and 0.0588775 in the MARS logit model. For the estimated MARS model the relative importance of T and NW was 100%, with the other three variables having 0%. The GCV is lowered to 1704 for the inclusion of both T and NW and 0.08529 for Y, LF and D2. The reason is the MARS logit model depends only on interactions between two variables, T and NW. The other variables were eliminated in the automatic model selection process as the software selects a simplified (less variables on the right) model. The MARS procedure eliminated D2 and Y and LF once the knot was discovered. These results are consistent with D2 being not identified in the usual probit specification.

The MARS analysis, by suggesting that there are knots in the model, gives us some insight into what is probably happening. By forcing the probit model into the constant coefficient assumption and ignoring interactions and adding possibly redundant variables, the researcher may be setting the stage for various nonlinear estimation problems. The fact that these problems are related to the convergence tolerance is of some interest to those working in this field.

7. Conclusions

What started as a replication of probit results reported in Maddala [16] was extended when different results were obtained with the software systems B34S, SAS, RATS, LIMDEP and Stata. The differences were traced to the effect of alternative convergence tolerances on both the rank of the variance covariance matrix and the point estimate of one variable that was caused by Quasicomplete Separation. Only two software systems, Stata and TSP, were able to detect the underlying problem that for this model and these data, the maximum likelihood estimate does not exist, Using a Multivariate Adaptive Regression Splines (MARS) model to relax the assumption of constant coefficients, further implications are drawn from this dataset.

A number of important conclusions emerge from this brief analysis. The most important is that the user should not just accept output uncritically, but probe the solution offered by the computer. It is a good idea to try nonlinear models on a variety of software systems. The experiment will either confirm your original findings or assist in uncovering something that would not have been found otherwise. The sensitivity of the results to the convergence tolerance in software systems suggests great awareness of the implications of these often hidden assumptions of estimation, which are usually ignored. It is important to experiment with the default settings when running nonlinear models. While Complete and Quasicomplete Separation and other problems are known, even experts in the probit/logit field such as Maddala can be fooled if the software system appears to converge and the coefficients and SE's errors appear reasonable. To guard against publishing flawed results it is highly recommended that the condition of the variance covariance matrix or the hessian matrix from which it is derived be available for inspection and possible reporting along with the log likelihood function. The gradient, which is often not available, gives indications concerning whether false convergence has occurred.

It may be the case that the underlying model is not a constant coefficient probit model, no matter what variables are placed on the right. The MARS analysis of the data in this paper suggests that the constant coefficient assumption implicit in the traditional probit model was not met. A simplified but less restrictive model was shown to outperform the more complex (more variables on the right) probit model. The results reported here suggest more analysis is needed of this interesting, real-world example since it appears to contain subtle numerical problems that are not readily apparent to most software systems.

Acknowledgement

Many suggestions received by B.D. McCullough and Charles Renfro are most appreciated and have helped to strengthen the paper in many ways. Evelyn Lehrer read an earlier draft and contributed in a number of ways. Diana A. Stokes provided editorial help. Any remaining errors are my responsibility.

References

[1] A. Albert and J. Anderson, On the existence of maximum likelihood estimates in logistic regression models, *Biometrika* **71** (1984), 1–10.

[2] T. Amemiya, *Advanced Econometrics*, Harvard University Press, Cambridge, MA, 1985.

[3] E. Anderson et al., *LAPACK Users' Guide*, Third Edition, Siam, Philadelphia, 1999.

[4] A.J. Barr, J.H. Goodnight, J.P. Sall and J.T. Helwig, *A User's Guide to SAS 76,* SAS Institute, Raleigh NC, 1976.

[5] T. Coleman and M. Branch and A. Grace, *Optimization Toolbox: For Use with Matlab*(*r*), The MathWorks, Natick, MA, 1999

[6] R. Davidson and J. MacKinnon, *Econometric Theory and Methods*, Oxford University Press, New York, 2003.

[7] J. Dongarra, C.B. Moler. J.R. Bunch and G.W. Stewart, *LINPACK Users' Guide*, Siam, Philadelphia, 1979.

[8] T. Doan, *RATS User's Manual*, Estima, Evanston, IL, 1992.
[9] J. Friedman, Multivariate Adaptive Regression Splines, *Annals of Statistics* **19**(1) (1991), 1–67.
[10] W. Gould and William Sribney, *Maximum Likelihood Estimation with Stata*, Stata Corporation, College Station, TX, 1999.
[11] W. Greene, *LIMDEP Version 7.0*, Econometric Software, 1998.
[12] B. Hall and C. Cummins *TSP Reference Manual, version 4.5*, TSP International, Palo Alto, CA, 1999.
[13] S. Kawasaki, Applications of Log-Linear Probability Models in Economics. Ph.D. dissertation, Northwestern University, 1979.
[14] J. Longley, An Appraisal of Least Squares Programs for the Electronic Computer from the Point of View of the User, *Journal of the American Statistical Association* **62**(319) (1967), 819–841.
[15] J. Longley, *Least Squares Computing Using Orthogonalization Methods*, Marcel Decker, New York, 1984.
[16] G.S. Maddala, *Introduction to Econometrics*, MacMillian, New York, 1988, 1992.
[17] B.D. McCullough and C.G. Renfro, Some Numerical Aspects of Nonlinear Estimation, *Journal of Economic and Social Measurement* **26** (2000), 63–77.
[18] B.D. McCullough and H.D. Vinod, The Numerical Reliability of Econometric Software, *Journal of Economic Literature* **37** (June 1999), 633–665.
[19] R. McKelvey and W. Zavoina, An IBM Fortran IV Program to Perform N-Chotomous Multivariate Probit Analysis, *Behavioral Science* **16** (March 1971), 186–187.
[20] W.S. McManus, Estimates of the Deterrent Effect of Capital Punishment: The Importance of the Researcher's Prior Beliefs, *Journal of Political Economy* **93** (April 1985), 417–425.
[21] V. Numerics, *IMSL Stat/Library and Math/Library*, IMSL, Houston, TX, 1987.
[22] J. Press and M. Nerlove, *Univariate and Multivariate Log-Linear and Logistic Models*, RAND Corp., Santa, Monica, CA, Report R-1306-EDA/NIA 1973.
[23] Quantitative Micro Software, *EViews version 3 User's Guide*, Quantitative Micro Software, 1997.
[24] H.H. Stokes, *Specifying and Diagnostically Testing Econometric Models*, Quorum Press, Westport, CT, 1991, 1997.
[25] StataCorp, *Stata Statistical Software Release 7.0*, Stata Corporation, College Station, TX, 2001.
[26] K. White, *SHAZAM version 8.0 User's Reference Manual*, McGraw-Hill, New York, 1997.