PANTOB INSTRUCTIONS

by Jeffrey R. Campbell and Bo E. Honoré, December 1991.

Pantob is a Gauss program which estimates a censored Tobit model with panel data as in Honoré (1992). The program runs with Gauss version 2.2. It requires that the Gauss module Optmum be properly installed on your hard drive. Pantob provides three optimization algorithms of its own.

Installing Pantob.

The code needed to run Pantob is in the following files:

```
pantob.f
bopt.rex
bodpt.rex
boddpt.rex
bocalv.rex
bosort.rex
```

The first file should be made part of your Gauss program using the `#include` statement. At the top of your program, put the following lines of code.

```
library optmum;
#include optmum.ext;
#include pantob.f;
```

If you wish to access other libraries in your program, you should include them in one `library` statement with `optmum`.

The remaining files, those with the extention "`.rex`", should be placed in the subdirectory `C:\ GAUSS \ GXE`. These are compiled Fortran routines which do the bulk of the work for Pantob. These are automatically loaded when you call the Pantob procedure. Once they are in the proper subdirectory, you need not worry about them.

Pantob uses Optmum to maximize the objective function. Pantob has three optimization routines built into it, the downhill simplex method, Powell's method, and the conjugate gradient method of Polak and Ribiere. These routines are procedures in the file `pantob.f`. See Press, Flannery, Teukolsky, and Vetterling (1986) for the details of these procedures. Pantob will run with no problem if you do not have Optmum, however you may only use the three methods for your optimization.

Running Pantob

To estimate a tobit model with fixed effects, you need only call the procedure `pantob`. Below is a sample program which estimates such a model using the Gauss data set "labordat". The first column of the data set is the dependent variable. The second contains an identification number. The third contains the only regressor. The optimization routines require starting values of the parameters being estimated. In this case, the routine starts from zero.

```
library optmum;
#include optmum.ext;
#include pantob.f;
{b,v,f,infile,met,yname,iname,xnames}=pantob("laborda t",1,2,3,0)
```

This program uses the default values of all global variables. In general, this is unwise.

Inside Pantob

When your program calls the `pantob` procedure, it first loads and sorts the given data set. It sorts the data according to the identification number and deletes all observations with missing data in any column used for the estimation. This reduces the amount of data processing you need to do in Gauss, but it may become a large computational burden if you repeatedly use the same data set. If you have a particularly large data set, you may want to alter the code so that this step is only performed once, at the beginning of the program.

After the data is processed, the program minimizes the chosen objective function over R^k where k is the number of regressors. The global variables `_ptloss` and `_pttheta` control the choice of an objective function. At their default values, they chose the quadratic loss function. The Optmum module gives you five algorythims for minimizing this function, and Pantob gives you three more. The global variable `_ptmet` controls the choice of an optimization algorythim. Optmum uses the global variable `__output` to control wheter running time output from the optimization is sent to the screen. The running time output generated by the methods Pantob provides is also controled by this global.

As a final step, the program estimates the variance covariance matrix of the estimator. V, the variance of the first derivative of the objective function, is always estimated using analytic derivatives. The global variable `_ptbw` controls the estimation of Γ. If `_ptbw` > 0, the estimatior uses numerical derivatives. If `_ptbw` $= 0$, the estimatior will use analytic derivatives if that is possible.

Your Data

Pantob assumes that your data is in a Gauss data set which is on the current directory. To convert your data from an ASCII file into a Gauss data set, you can use the Atog data utility. See the Gauss documentation for more information on Atog. You will need to tell Pantob the column numbers of the data set which contain your dependent variable, identification number, and regressors.

If your data set has missing data for some observations, this will not cause a problem. Pantob will automatically delete observations with missing data in any relevant column. After this operation, the data set may not be a true panel. If there are individuals with only one observation, Pantob will ignore them during estimation. If either of these steps leaves you with no observations, Pantob will tell you this and stop.

Honoré(1992) contains two rank assumptions which your data must fulfill. Assumption R.1 specifies that ΔX is of full rank. Variables which do not change over the panel for a given individual do not fulfil this requirement. This excludes many variables of interest such as race and gender. Any such variable is " absorbed into the fixed effect ". Assumption V requires that the covariance matrix of the objective function's first derivative and the objective function's second derivative are of full rank when evaluated at the true value of β. If your data satisfy assumption R.1., the covariance matrix of the first derivative, V, should not present a problem. The second derivative, Γ may still not be of full rank, particularly if you are using the absolute value or polynomial loss functions. In these cases, Γ is of full rank only if $E[\Delta X | \Delta Y - \Delta X\beta \in O]$ is of full rank where O is an open ball of arbitrarily small diameter around zero. Your data should satisfy the assumption if this holds for some "small" ball around zero. If the assumption does not hold, then your estimate, $\hat{\Gamma}$, will fail to invert. If this occurs, Pantob will print $\hat{\Gamma}$, dump it to the current auxiliary output, and then crash.

Analysis of this matrix can often reveal what is wrong with your data. If one column of $\hat{\Gamma}$ contains only zeros, the regressor corresponding to it is not of full rank. If $\hat{\Gamma}$ is a matrix of zeros, you are probably not having a problem with your data, but rather a problem with estimating Γ. See the section below on estimating Γ before you conclude that your data violate assumption V.

The simplist way to include transformations of your data in the estimation is to form a new data set by appending the transformations onto the original data set. For example, if `age` was one of your dependent variables and you wanted to include its square in the estimation, you should write a program like this:

```
@ addsq.gsp @
infile="agedat; outfile="agedat2";
open f1=^infile;
varnames=getname(infile);
sel=(varnames.$=="age");
create fout=^outfile with ^varnames,0,8;

do until eof(f1);
x=readr(f1,100);
agesq=(x*sel)^2;
x=x~agesq;
call writer(fout,x);
endo;

. closeall;
```

Be wary of taking the log of your dependent variable. If you believe that the dependent variable's level is generated by a tobit model, it is not at all clear this model also applies to its log.

Loss Functions

For a given symmetric convex function $\Xi(d)$ with derivative $\xi(d)$ define the function $s(y_1, y_2, \delta)$ as follows:

$$s(y_1,y_2,\delta)=\begin{cases}\Xi(y_1)-(y_2+\delta)\xi(y_1) & \text{if } \delta\le -y_2;\\ \Xi(y_1-y_2-\delta) & \text{if } -y_2<\delta<y_1;\\ \Xi(-y_2)-(\delta-y_1)\xi(-y_2) & \text{if } y_1\le\delta.\end{cases}$$

If $\Xi(d) = d^2$ then $s(y_1, y_2, \delta) = \chi(y_1, y_2, \delta)$ and if $\Xi(d) = |d|$ then $s(y_1, y_2, \delta) = \varphi(y_1, y_2, \delta)$ where χ and φ are defined in Honoré (1992). Minimization of the mean of φ across the sample yields the estimator $\hat{\beta}_3$. Similarly, minimization of the mean of χ yields $\hat{\beta}_4$. Each choice of a loss function, $\Xi(d)$ yields a different estimator for the tobit model with fixed effects. Consider the following loss function:

$$\Xi(d,\theta)=\begin{cases}15(\theta d)^2-5(\theta d)^4+(\theta d)^6 & \text{for -1} \le \theta d\le 1\\ 11-16(1+\theta d) & \text{for } \theta d\le -1\\ 11+16(\theta d-1) & \text{for } 1\le\theta d\end{cases}$$

For a given value of θ, this loss function also defines an estimator. This function is the *polynomial* loss function. The parameter θ indexes it. This is a C^2 function, so the matrix Γ can be computed analytically in stead of numerically. Note that $lim_{\theta\to 0}\frac{\Xi(d,\theta)}{15\theta^2} = d^2$ and that $lim_{\theta\to\infty}\frac{\Xi(d,\theta)}{16\theta} = |d|$. The polynomial loss function is just a combination of the quadratic and absolute value loss functions. In a sense, it is superior to either one of them alone. It combines the differentiability of the quadratic with the robustness of the absolute value.

Pantob allows you to chose between estimators defined by the quadratic, absolute value, or polynomial loss functions. The global variable `_ptloss` governs this choice. If you select the polynomial loss function, set `_pttheta` equal to your desired value of θ.

Requirements the estimation imposes on the data will help you select a loss function. If your dependent variable has a high censoring probability, assumption V will be difficult to fulfill in practice. Assumption V places the fewest demands on the data when the quadratic loss function defines the estimator. On the other hand, if you feel that outliers are a problem with your data set; then the robustness of the absolute value loss function is desirable. The polynomial loss function allows you to trade off between these two extremes.

Estimating Γ

Pantob implements the estimators of Γ proposed in Honoré (1992). Estimators based on numerical derivatives are available for use with any of the loss functions. The global variable `_ptbw` controls the bandwidth used. This may be either a scaler or a vector. If it is a scaler, the estimator uses the same bandwidth when taking the derivative with respect to each of the coefficients. If it is a vector, the bandwidth for the derivative with respect to the jth coefficent is its jth element.

For the quadratic and polynomial loss functions, estimators based on analytic derivatives are also available. Setting `_ptbw`$= 0$ selects these estimators.

Estimating Γ is probably the trickiest part of the estimation procedure. With the quadratic loss function, this usually presents no problem. A convenient analytic expression for the second derivative of the loss function exists, so it's expectation is estimated with a mean. No such expression exists for the absolute value loss function.

Let $f(y_1, y_2, \Delta X\beta)$ be the first derivative of $s(y_1, y_2, \Delta X\beta)$. Then Γ is defined by

$$\Gamma = \frac{\partial E[f(y_1, y_2, \Delta X\beta_0)]}{\partial\beta}$$

To estimate this for the absolute value loss function, the partial derivative is replaced by a finite difference and the expectation is replaced by taking the mean over the sample.

This estimation procedure has an inherent variance-bias trade off. A larger bandwidth reduces the variance while increasing the bias. With the absolute value loss function, the only observations which affect $\hat{\Gamma}$ are those for which $|\Delta Y - \Delta X\beta| <$`_ptbw`. This is clear after inspection of the expression for $\hat{\Gamma}_3^{(ij)}$ in Honoré (1992). If the bandwidth is small, the finite difference approximates the derivative well; and the bias of $\hat{\Gamma}$ declines. A small bandwidth also reduces the number of observations being used in the mean, so the variance of $\hat{\Gamma}$ increases.

A high censoring probability for the dependent variable can cause problems with estimating Γ. Only observations for which either $y_1 > 0$ or $y_2 > 0$ affect $\hat{\Gamma}_3$. Imposing this requirement along with that above may yield an estimate which is affected by only a handful of observations. If the total number of observations is large, $\hat{\Gamma}_3$ will be very close to zero.

Unfortunately, using a polynomial loss function and letting θ be large is not a solution for these problems. Although analytic derivatives may be used to estimate Γ, the only observations which will affect the estimate are those for which either $y_1 > 0$ or $y_2 > 0$ and $|\Delta Y - \Delta X\beta| < 1/\theta$. In this case, $1/\theta$ plays the same role as `_ptbw` does above. They both define how close to zero an observation must be to affect $\hat{\Gamma}$. Reducing $1/\theta$ does not reduce the bias of the estimator, because it uses analytic derivatives, although it will increase its variance. A high censoring probability will also cause problems with this estimator if θ is large.

Optimization

All of the estimators for the tobit model with fixed effects are defined as minimizers of criterion functions. For this task, Pantob provides three routines, the downhill simplex method, Powell's method, and the Polak-Ribiere conjugate gradient method. Pantob also allows you to use the five routines contained in the Optmum module. The global variable `_ptmet` controls the choice of optimization routine. To send running time output from any of the routines, set the global variable `__output`= 2. Otherwise, set `__output`= 1; This is the case whether you are using the routines provided by Pantob or those from Optmum.

When using one of the Optmum routines, you may use any of the global variables discussed in the Optmum documentation except `_opalgr` to control the optimization. `_opalgr` is superseded by `_ptmet`. The global variables `_amftol` and `_amitmax` control the operation of the routines provided by Pantob. All of the methods will stop if the number of iterations exceeds `_amitmax`. The downhill simplex method will stop when the difference of the function value at the high point of the simplex and that at the low point is less than `_amftol`. The other two routines will stop when the function value fails to improve by `_amftol`. The global variables `_amalp`, `_ambet`, and `_amgam` control the downhill simplex method. All of the optimizers Pantob provides are adapted from Press, Flannery, Teukolsky, and Vetterling (1986). See there for the details of their operation.

No matter which routine you use, it is always a good idea to restart the optimization from the point where it converged. If your result does not change much, then you can be more confident that the point is a minimum and not a saddle point or ridge. This is especially good advice if you are using any of the methods Pantob provides. To restart the optimization, simply call Pantob a second time using the parameter estimate from the first estimation as the starting value. It is advisable to restart the optimization using a different optimization method. We have encountered situations in practice where no single optimization method finds a minimum, but combining two of them does quite well.

If your starting value is poor, the routines in Optmum may never find the minimum of the objective function. In this case, the downhill simplex method will locate a neighborhood of the minimum relatively rapidly. One of the other routines can then pinpoint the minimum. The demonstration program `demo.gsp` pursues this strategy.

Optimization can be more difficult when the estimator uses the absolute value loss function. In this case, the second derivative of the objective function may be zero in a neighborhood of the minimum. Optimization methods which set the gradient of the objective function equal to zero will generally crash when this happens, because its Hessian is non-invertable. All of the routines provided by Optmum have this problem. Even if this is a problem, these methods may still be useful. They may find a neighborhood of a minimum before crashing. One of the other minimizers may be started from the point it crashed at to refine the minimum. This point is stored in the global variable `_opparnm`.

Using WRTPT

The file `wrtpt` contains a procedure for displaying the results generated by Pantob. The procedure, wrtpt, takes all of the output from Pantob as input and returns nothing. To use the procedure, use the `call` command.

```
call wrtpt(b,v,f,met,yname,iname,xnames,colour)
```

Pantob generates all of the inputs except `colour`. Set this equal to 1 if you do not have a color monitor. Set it greater than one otherwise. To use the procedure, `#include` the files `wrtpt` and `colors` at the top of your program.

Wrtpt reports the coefficient estimates, their standard errors, and their t-statistics. It also reports the loss function used, the optimization method, the dependent variable, the identification variable, and a χ^2 test for joint significance of the coefficients.

To send the screen to an output file, use the `screen out` command. See the demonstration program for an example.

Demonstration Program

The file `demo.gsp` is a Gauss program which illustrates the use of the Pantob module. The program's comments should make its content self explanitory.

`demo.gsp` uses the Gauss dataset `baddat`. This is simulated data, but it is an unbalanced panel with missing observations. Pantob automatically handles these two common data problems. The model generating the data is

$$\begin{aligned} Y_{it}^{\star} &= \alpha_i + X_{it}\beta + \varepsilon_{it} \\ Y_{it} &= \max(0, Y_{it}^{\star}) \\ \beta &= (1,1,0,0,0)' \end{aligned}$$

Where X_{it} is a 1×5 vector. The first column of the data set is the dependent variable, Y_{it}. The second column is the identification number, i. The third through eighth columns are the componants of the vector X_{it}.

There are 5000 individuals in the data set. The number of observations for each individual is uniformly distributed between 1 and 5. The probability that any given element of the data matrix is missing is $\frac{1}{10}$. Identification numbers are never missing. Although none of the identification numbers are missing for this data set, Pantob can handle data with this problem. It deletes all observations with no identification number.

The demonstration program estimates the same model using each of the available loss functions. It uses the procedure `wrtpt` to write the output to the screen, and it saves the output in the file `demo.out`. The program impliments much of the advice offered above, particularly that regarding optimization.

Pantob Global Variables

Four global variables control the operation of Pantob, and five control Amoeba. Two others are used internally by it and so should not be used in your program. The global variables are:

`_ptloss` This controls your choice of a loss function for the estimation. This impliesa choice of an estimator. Your choices are:

1	Quadratic loss function
2	Absolute value loss function
3	Polynomial loss function (See `_pttheta` below)

Default value = 1

`_pttheta` This is your choice of θ for the polynomial loss function. See the section on loss functions above for details.

Default value = 3

`_ptbw` This is the band width used when estimating Γ with numerical derivatives. If it is a scaler, all of the derivatives will use the same band width. If it is a vector, its jth element is the bandwidth for the derivative with respect to the jth coefficient. If it is set equal to zero, the estimator will use analytic derivatives if this is possible. Analytic derivatives may not be used if you are estimating with the absolute value loss function. In this case, if it is set equal to zero, Pantob will warn you than you have made a mistake and set it equal to $\frac{1}{8}$.

Default value = 0;

`_ptmet` This controls the optimization method used to minimize the loss function. Your choices are:

-2	Polak-Ribiere Conjugate Gradient method
-1	Powell's method
0	Downhill simplex method
1	Steepest descent
2	Broyden,Fletcher,Goldfarb, Shanno method(BFGS)
3	Scaled BFGS
4	Davidon,Fletcher,Powell method
5	Newton-Raphson method

See Press, Flannery, Teukolsky, and Vetterling (1986) for descriptions of the first three methods. The other methods are the routines found in the Optmum module. If one of these methods is selected, the value of `_ptmet` is passed the global variable `_opalgr` used by Optmum to select an optimization algorithm. The Gauss applications manual contains details of these algorithms.

Default value = 1

`_amftol` This sets the convergence tolerance for the methods Pantob provides.

Default value = 10^{-4}

`_amalp` This is the first extrapolation constant for the simplex method.

Default value = 1

`_ambet` This is the contraction constant for the simplex method.

Default value = $\frac{1}{2}$

`_amgam` This is the second extrapolation constant for the simplex method.

Default value = 2

`_amitmax` This is a ceiling over the number of iterations the methods Pantob provides will apply to a problem.

Default value = 200

See below for details on the function of these global variables.

`xx,dat1` These globals are used to store the data. They should not be used in your program.

PANTOB

Purpose

Estimates a tobit model with panel data using the techniques in Honoré(1992).

Format

{*b,v,f,infile,met,yname,iname,xnames*}=**Pantob**(*dset,yvar,ivar,xvars,sval*)

Input

dset	String containing the name of a **Gauss** dataset
yvar	integer, column number of the dependent variable
ivar	integer, column number individual reference variable
xvars	$K \times 1$ vector of integers, column numbers of the regressors
sval	$K \times 1$ vector, starting values for minimization routine

Output

b	$K \times 1$ vector, estimates of regression coefficients
v	$K \times K$ matrix, estimated variance/covariance matrix of b
f	scaler, criterion function evaluated at b
infile	string, identical to the input “*dset*
met	scaler, value of `_ptmet` during estimation
yname	string, label of dependent variable.
iname	string, label of identification variable.
xnames	$K \times 1$ string vector, labels of regressors.

WRTPT

Purpose

Displays the output of **Pantob**

Format

WRTPT(*b,v,f,infile,met,yname,iname,xnames,colour*)

Inputs

b	$K \times 1$ vector, estimates of regression coefficients
v	$K \times K$ matrix, estimated variance/covariance matrix of b.
f	scaler, criterion function evaluated at b
infile	string, identical to the input "*dset*
met	scaler, value of `_ptmet` during estimation
yname	string, label of dependent variable.
iname	string, label of identification variable.
xnames	$K \times 1$ string vector, labels of regressors.
colour	scaler, set *colour*>1 if you have a color monitor. Set it equal to one otherwise.

References

Press, W. , B. Flannery, S. Teukolsky, and W. Vetterling (1986): *Numerical Recipes: The Art of Scientific Computing.* Cambridge: Cambridge University Press.

Honoré, B.E. (1992) "Trimmed LAD and Least Squares Estimation of Truncated and Censored Regression Models with Fixed Effects," accepted for publication, *Econometrica.*