Soc. 740
J. Fox

Additions to Arc

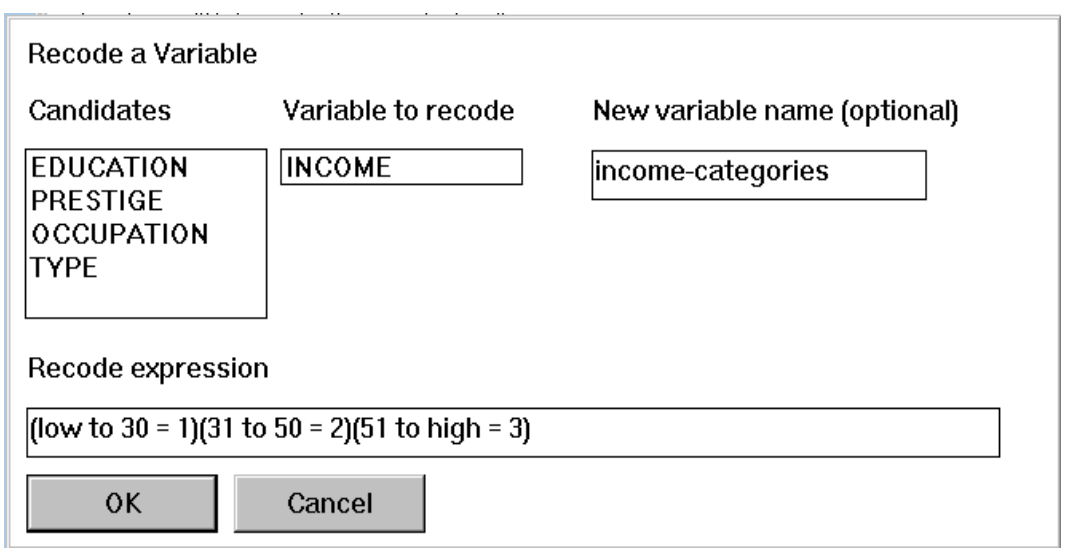
This handout describes a few small additions to Arc. All of the functions and methods that are meant to be used directly are documented through the Lisp-Stat help facility. In most cases, you will simply use these functions and methods through Arc menus and dialogs.

- Regression models have a new *F-tests* menu item. If you select this item, an analysis-of-variance table is printed for the model, testing the hypothesis that each term in the model is zero. The method invoked will correctly handle terms with several degrees of freedom (e.g., in dummy-variable regression). A similar *Wald Tests* menu item is provided for generalized linear models, such as logistic regression.

  You can also invoke the following methods directly by sending messages to regression-model or generalized-linear-model objects:

`:TEST-LINEAR-HYPOTHESIS`	Provides an F-test or a Wald test for any linear hypothesis
`:F-TESTS`	Prints an analysis-of-variance table for a regression model
`:WALD-TESTS`	Prints a table of Wald tests for a generalized linear model

- The menu for a dataset has a new *Recode variable* menu item which opens a recode dialog. This dialog performs the same function as the `recode` statement in SPSS or the `code` statement in Minitab:



  In this example, I double-clicked on `INCOME` in the *Candidates* box to transfer it to the *Variable to recode* box; I typed the name `income-categories` into the *New variable*

name box. Had I not specified a new variable name, `INCOME` would have been replaced with the recoded version of the variable, after asking for confirmation. The recode specification has this general structure:

(*old value, list of old values, or range of old values = new value*)

- Values in a list are separated by one or more blanks; e.g., `(1 5 3 6 = "white collar")`. You can also use commas to separate values. You do not need blanks around the equal sign.

- Notice that you can specify text as well as numeric values for variables, both as old and new codes, and that text values can contain blanks. Enclose all text values in double-quotes.

- You can also recode to and from missing data, representing missing data with a question mark; e.g., `(0=?)`. Do not enclose the missing-data question mark in quotes.

- Ranges of values are specified by using `to` or a colon (`:`); e.g., `(1 to 5 = 1)` or `(1:5=1)`.

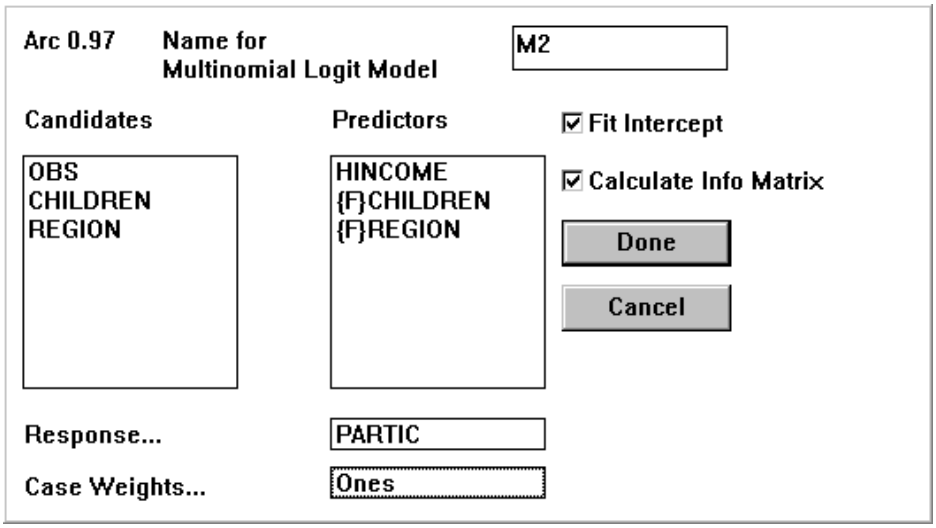
- The last recode specification can be of the form `(else = value)`, e.g., `(else=3)`.

- You can use the special values `low` or `lo` and `high` or `hi` to represent the minimum and maximum values of a variable. Alternatively, if you omit one end of a range, the low or high value is implied; e.g., `(to 10 = 1)` implies `(low to 10 = 1)`; and `(10:=2)` implies `(10:high=2)`.

- You can use an arrow (`->`) in place of =; e.g., `(low to 10 -> 1)` is the same as `(low to 10 = 1)`.

- Adapting a Lisp-Stat program written by Jan DeLeeuw at UCLA, I've added a method for displaying axes representing principal components in three-dimensional spinning plots. Send the message `:pca` to the plot to display the principal components; send the message `:pca :erase t` to erase the principal components. For example, `(send plot1 :pca)`. You need to type the message in the Listener window.

- I've added a program for fitting polytomous logit models to Arc. The program is accessed by selecting *Fit multinomial logit model...* from the *Graph&Fit* menu, producing the following dialog box:



The response variable can be a numeric or text variable, giving the response category for each observation. If the data represent individual observations (the most common case), then *Case Weights...* should be set to `Ones`. You can make the program run a little faster by unchecking *Calculate Info Matrix*. Coefficient standard errors and Wald test will then be a little less accurate.

You can input data from a contingency table, by including the count for each cell of the table in a variable (called, e.g, `count`). Here, for example, is how Table 15.8 on page 487 of the text could be input, adding 0.5 to each cell count because of the zero cell:

Race	Nonwhite	Law	Voting	Count
White	<30	Restrictive	All/Most	108.5
White	<30	Restrictive	Some	58.5
White	<30	Restrictive	Never	64.5
White	<30	Moderate	All/Most	190.5
White	<30	Moderate	Some	74.5
White	<30	Moderate	Never	108.5
White	>30	Restrictive	All/Most	46.5
White	>30	Restrictive	Some	18.5
White	>30	Restrictive	Never	21.5
White	>30	Moderate	All/Most	41.5
White	>30	Moderate	Some	10.5
White	>30	Moderate	Never	16.5
Black	<30	Restrictive	All/Most	3.5
Black	<30	Restrictive	Some	5.5
Black	<30	Restrictive	Never	5.5
Black	<30	Moderate	All/Most	15.5
Black	<30	Moderate	Some	8.5
Black	<30	Moderate	Never	26.5
Black	>30	Restrictive	All/Most	0.5
Black	>30	Restrictive	Some	6.5
Black	>30	Restrictive	Never	76.5
Black	>30	Moderate	All/Most	3.5
Black	>30	Moderate	Some	10.5
Black	>30	Moderate	Never	27.5

Notice that this differs from how you would enter data for a binomial logit model, where, for each combination of independent variables, you input the total count and the number of 'successes.'

The polytomous logit program behaves more or less like the other fitting methods in Arc, returning a model menu, which, however, includes only three items:

- *Summary* reprints the summary report for the model. This is useful, for example, if you delete some observations and want to see the effect.

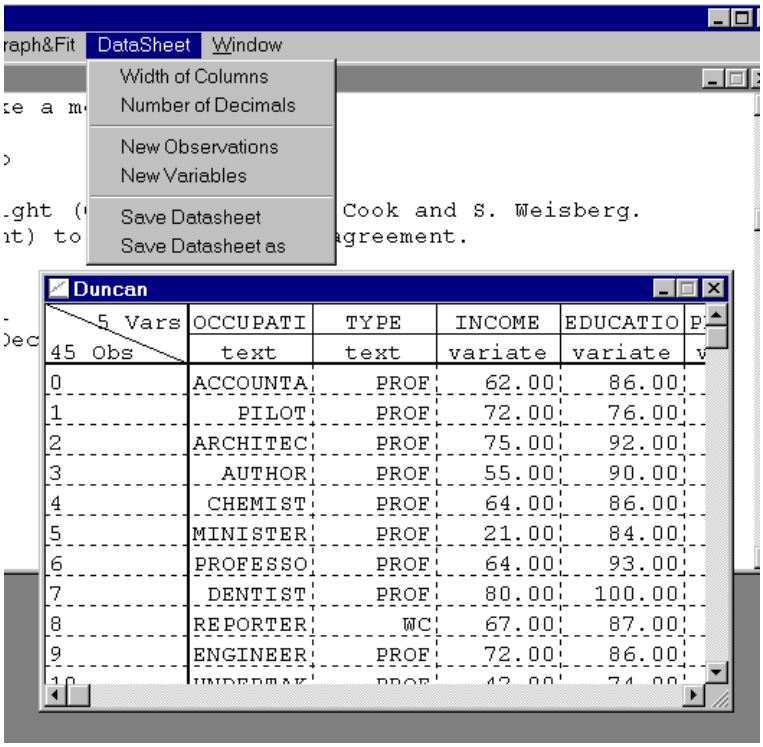
- *Wald tests* prints tests for all of the terms in the model, as for binomial logit models.

- *Remove model* does the obvious.

- Having fit the model, case-wise components of the deviance are added to the dataset (e.g., `M1:Dev-Components`). These can be plotted against case numbers to see which observations contribute most to the deviance.

Note: This program does not employ a true multinomial logit model, but rather fits an equivalent polytomous logit model in which observations are weighted by the case weights. As a consequence, the deviance for a 'saturated' model is not 0. Coefficients, standard errors, and *differences* in the deviance between models are, however, identical. See the analogous discussion of the binary vs. binomial logit model in Section 15.3.1 (pp. 482-484) of the text.

- I have adapted Lisp-Stat code written by Forrest Young (for his ViSta system) to provide a data spreadsheet for Arc datasets. To view the spreadsheet for a dataset, click *Show data sheet* in the dataset's menu. When the window with the datasheet has the focus, a *DataSheet* menu will be installed in the menu bar. Click on this menu to see the (self-explanatory) actions that you can perform. To open the data spreadsheet to modify or add data, click *Edit data sheet* in the dataset's menu. You can change a data value by clicking in a cell and typing the new value. You can add new rows or columns to the dataset either by clicking the appropriate controls in the datasheet window or by using the corresponding items in the *DataSheet* menu. After modifying a datasheet, you need to save it in order to incorporate the changes in the corresponding Arc dataset. Alternatively, you can save the datasheet to a new Arc dataset. Likewise, if you make a change to the Arc dataset (e.g., by recoding a variable), you need to show or edit the datasheet again to see the changes. Finally, by clicking *New Data* under the *Arc* menu, you can enter data into an empty datasheet, creating a new Arc dataset.



Datasheets only display numeric and text variables, not factors or interactions. Navigate around the datasheet by using the standard windows scrollbars.

In addition to these new facilities for Arc, I've written the following Lisp-Stat functions:

- The function `Box-Tidwell` returns maximum-likelihood estimates for the transformation parameters in the Box-Tidwell regression model.

  The general syntax for `Box-Tidwell` is as follows, where italic iterms are replaced as appropriate:

  ```
  (box-tidwell (list list-of-xs-to-transform) y
               :other-x (list list-of-other-xs)
               :tol tolerance   :score-test t-or-nil)
  ```

  where:

list-of-xs-to-transform	is a list of independent variables that are candidates for transformation.
y	is the dependent variable.
list-of-other-xs	is a list of independent variables that are not candidates for transformation; the default is `nil` (i.e., no other variables).

tolerance	is a convergence criterion; when the maximum change in the transformation estimates is less than 100 tol %, iterations cease; default, .01.
t-or-nil	if `nil` suppresses the score tests for the transformation parameters; default, `t`.

Example:

```
(box-tidwell (list income education) prestige
        :other-x (list percent-women (^ percent-women 2)))
```

- The function `Box-Cox-constructed-variable` returns the constructed variable for the Box-Cox transformation of the dependent variable in regression. The syntax for `Box-Cox-constructed-variable` is:

  (`box-cox-constructed-variable` *y*)

  where *y* is the dependent variable.

  You can use this function in the Arc *add a variate* dialog.