$$\ln\left\{\frac{\Pr(y=m\mid \mathbf{x})}{\Pr(y>m\mid \mathbf{x})}\right\} = \tau_m - \mathbf{x}\boldsymbol{\beta} \quad \text{for } m=1 \text{ to } J-1$$

where the $\boldsymbol{\beta}$s are constrained to be equal across outcome categories, whereas the constant term τ_m differs by stage. As with other logit models, we can also express the model in terms of the odds:

$$\frac{\Pr(y=m\mid \mathbf{x})}{\Pr(y>m\mid \mathbf{x})} = \exp(\tau_m - \mathbf{x}\boldsymbol{\beta})$$

Accordingly, $\exp(-\beta_k)$ can be interpreted as the effect of a unit increase in x_k on the odds of being in m compared with being in a higher category given that an individual is in category m or higher, holding all other variables constant. From this equation, the predicted probabilities can be computed as

$$\Pr(y=m\mid \mathbf{x}) = \frac{\exp(\tau_m - \mathbf{x}\boldsymbol{\beta})}{\prod_{j=1}^{m}\{1+\exp(\tau_j - \mathbf{x}\boldsymbol{\beta})\}} \quad \text{for } m=1 \text{ to } J-1$$

$$\Pr(y=J\mid \mathbf{x}) = 1 - \sum_{j=1}^{J-1}\Pr(y=j\mid \mathbf{x})$$

These predicted probabilities can be used for interpreting the model. In Stata, this model can be fitted using `ocratio` by Wolfe (1998); type `net search ocratio` and follow the prompts to download.

6 Models for nominal outcomes with case-specific data

An outcome is nominal when the categories are assumed to be unordered. For example, marital status can be grouped nominally into the categories of divorced, never married, married, or widowed. Occupations might be organized as professional, white collar, blue collar, craft, and menial, which is the example we use in this chapter. Other examples include reasons for leaving the parents' home, the organizational context of scientific work (e.g., industry, government, and academia), and the choice of language in a multilingual society. Further, in some cases a researcher might prefer to treat an outcome as nominal, even though it is ordered or partially ordered. For example, if the response categories are strongly agree, agree, disagree, strongly disagree, and don't know, the category "don't know" invalidates models for ordinal outcomes. Or, you might decide to use a nominal regression model when the assumption of parallel regressions is rejected. In general, if you have concerns about the ordinality of the dependent variable, the potential loss of efficiency in using models for nominal outcomes is outweighed by avoiding potential bias.

This chapter focuses on three closely related models for nominal (and sometimes ordinal) outcomes with case-specific data. The multinomial logit model (MNLM) is the most frequently used nominal regression model. In this model, you are essentially estimating a separate binary logit for each pair of outcome categories. Next we consider the multinomial probit model with uncorrelated errors, which is the normal counterpart to the MNLM. We then discuss the stereotype logistic regression model (SLM). Although this model is often used for ordinal outcomes, it is closely related to the MNLM. All these models assume that the data are case specific, meaning that each independent variable has one value for each individual. Examples of such variables are an individual's race or education. In the next chapter, we consider models that include alternative-specific data.

Models for nominal outcomes, both in this chapter and the next, require us to be more exacting about some basic terminology. Until now we have used "individual", "observation", and "case" interchangeably to refer to observational units, where each observational unit corresponds to a single row or record in the dataset. In the next two chapters, we will use only the term "case" for this purpose. Most of the time, we use the word "alternative" to refer to a possible outcome. Sometimes we refer to an alternative as an outcome category or a comparison group in order to be consistent with the usual terminology for a model or the output generated by Stata. The term "choice" refers to the alternative that is actually observed, which can be thought of as the "most preferred" alternative. For example, if the dependent variable is the party voted for in the last presidential election, the alternatives might be Republican, Democrat, and Independent. If the person corresponding to a given case voted for the alternative of Democrat, we would say that the choice for this case is Democrat. But you should not infer from the term "choice" that the models we describe can be used only for data where the outcome occurs through a process of choice. For example, if we were modeling the type of injuries that people (i.e., cases) entering the emergency room of a hospital have, we would use the term "choice" even though the injury sustained is unlikely to be a choice. We will continue with this terminology in chapter 7, but with one complication. Chapter 7 deals with alternative-specific variables that vary not only by case but also by the alternative. For example, if a commuter is selecting one of three modes of travel, an alternative-specific predictor might be her travel time using each alternative. Each case has three rows of data, one for each of the alternatives, since this is the easiest way to organize the data. We discuss this more fully in the next chapter.

We begin by discussing the MNLM, where the biggest challenge is that the model includes many parameters and it is easy to be overwhelmed by the complexity of the results. This complexity is compounded by the nonlinearity of the model, which leads to the same difficulties of interpretation found for models in prior chapters. Although fitting the model is straightforward, interpretation involves many challenges that are the focus of this chapter. We begin by reviewing the statistical model, followed by a discussion of testing, fit, and finally methods of interpretation. These discussions are intended as a review for those who are familiar with the models. For a complete discussion, see Long (1997). As always, you can obtain sample do-files and data files by downloading the spost9_do and spost9_ado packages (see chapter 1 for details).

6.1 The multinomial logit model

The MNLM can be thought of as simultaneously estimating binary logits for all comparisons among the alternatives. For example, let occ3 be a nominal outcome with the categories M for manual jobs, W for white-collar jobs, and P for professional jobs. Assume that there is one independent variable, ed, measuring years of education. We can examine the effect of ed on occ3 by estimating three binary logits,

$$\ln\left\{\frac{\Pr(P\mid \mathbf{x})}{\Pr(M\mid \mathbf{x})}\right\} = \beta_{0,P|M} + \beta_{1,P|M}\,\texttt{ed}$$
$$\ln\left\{\frac{\Pr(W\mid \mathbf{x})}{\Pr(M\mid \mathbf{x})}\right\} = \beta_{0,W|M} + \beta_{1,W|M}\,\texttt{ed}$$
$$\ln\left\{\frac{\Pr(P\mid \mathbf{x})}{\Pr(W\mid \mathbf{x})}\right\} = \beta_{0,P|W} + \beta_{1,P|W}\,\texttt{ed}$$

where the subscripts ~~to the βs~~ indicate which comparison is being made (e.g., $\beta_{1,P|M}$ is the coefficient for the first independent variable for the comparison of P and M).

The three binary logits include redundant information. Because $\ln a/b = \ln a - \ln b$, the following equality must hold:

$$\ln\left\{\frac{\Pr(P\mid \mathbf{x})}{\Pr(M\mid \mathbf{x})}\right\} - \ln\left\{\frac{\Pr(W\mid \mathbf{x})}{\Pr(M\mid \mathbf{x})}\right\} = \ln\left\{\frac{\Pr(P\mid \mathbf{x})}{\Pr(W\mid \mathbf{x})}\right\}$$

This implies that

$$\begin{aligned}\beta_{0,P|M} - \beta_{0,W|M} &= \beta_{0,P|W}\\ \beta_{1,P|M} - \beta_{1,W|M} &= \beta_{1,P|W}\end{aligned} \tag{6.1}$$

In general, with J alternatives, only $J-1$ binary logits need to be estimated. Estimates for the remaining coefficients can be computed using equalities of the sort shown in (6.1).

The problem with fitting the MNLM by estimating a series of binary logits is that each binary logit is based on a different sample. For example, in the logit comparing P with M, those in W are dropped. To see this, we can look at the output from a series of binary logits. First, we estimate a binary logit comparing manual and professional workers:

```
. use http://www.stata-press.com/data/lf2/nomintro2, clear
(1982 General Social Survey)
. tab prof_man, miss

   prof_man |      Freq.     Percent        Cum.
------------+-----------------------------------
     Manual |        184       54.60       54.60
       Prof |        112       33.23       87.83
          . |         41       12.17      100.00
------------+-----------------------------------
      Total |        337      100.00
. logit prof_man ed, nolog
Logistic regression                               Number of obs   =        296
                                                  LR chi2(1)      =     139.78
                                                  Prob > chi2     =     0.0000
Log likelihood = -126.43879                       Pseudo R2       =     0.3560

------------------------------------------------------------------------------
    prof_man |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          ed |   .7184599   .0858735     8.37   0.000      .550151    .8867688
       _cons |  -10.19854   1.177457    -8.66   0.000    -12.50632   -7.89077
------------------------------------------------------------------------------
```

Forty-one cases are missing for prof_man and have been deleted. These correspond to respondents who have white-collar occupations. Likewise, the next two binary logits also exclude cases corresponding to the excluded category:

```
. tab wc_man, miss

     wc_man |      Freq.     Percent        Cum.
------------+-----------------------------------
     Manual |        184       54.60       54.60
   WhiteCol |         41       12.17       66.77
          . |        112       33.23      100.00
------------+-----------------------------------
      Total |        337      100.00

. logit wc_man ed, nolog

Logistic regression                               Number of obs   =        225
                                                  LR chi2(1)      =      16.00
                                                  Prob > chi2     =     0.0001
Log likelihood = -98.818194                       Pseudo R2       =     0.0749

------------------------------------------------------------------------------
      wc_man |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          ed |   .3418255   .0934517     3.66   0.000     .1586636    .5249875
       _cons |  -5.758148   1.216291    -4.73   0.000    -8.142035   -3.374262
------------------------------------------------------------------------------

. tab prof_wc, miss

    prof_wc |      Freq.     Percent        Cum.
------------+-----------------------------------
   WhiteCol |         41       12.17       12.17
       Prof |        112       33.23       45.40
          . |        184       54.60      100.00
------------+-----------------------------------
      Total |        337      100.00

. logit prof_wc ed, nolog

Logistic regression                               Number of obs   =        153
                                                  LR chi2(1)      =      23.34
                                                  Prob > chi2     =     0.0000
Log likelihood = -77.257045                       Pseudo R2       =     0.1312

------------------------------------------------------------------------------
     prof_wc |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          ed |   .3735466   .0874469     4.27   0.000     .2021538    .5449395
       _cons |  -4.332833   1.227293    -3.53   0.000    -6.738283   -1.927382
------------------------------------------------------------------------------
```

The results from the binary logits can be compared with the output from `mlogit`, the command that fits the MNLM:

```
. tab occ3, miss

       occ3 |      Freq.     Percent        Cum.
------------+-----------------------------------
     Manual |        184       54.60       54.60
   WhiteCol |         41       12.17       66.77
       Prof |        112       33.23      100.00
------------+-----------------------------------
      Total |        337      100.00
```



```
. mlogit occ3 ed, nolog
Multinomial logistic regression                   Number of obs   =        337
                                                  LR chi2(2)      =     145.89
                                                  Prob > chi2     =     0.0000
Log likelihood = -248.14786                       Pseudo R2       =     0.2272

------------------------------------------------------------------------------
        occ3 |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
WhiteCol     |
          ed |   .3000735   .0841358     3.57   0.000     .1351703    .4649767
       _cons |  -5.232602   1.096086    -4.77   0.000    -7.380892   -3.084312
-------------+----------------------------------------------------------------
Prof         |
          ed |   .7195673   .0805117     8.94   0.000     .5617671    .8773674
       _cons |  -10.21121   1.106913    -9.22   0.000    -12.38072   -8.041698
------------------------------------------------------------------------------
(occ3==Manual is the base outcome)
```

The output from `mlogit` is divided into two panels. The top panel is labeled `WhiteCol`, which is the value label for the second category of the dependent variable; the second panel is labeled `Prof`, which corresponds to the third outcome category. The key to understanding the two panels is the last line of output: `occ3==Manual is the base outcome`. This means that the panel `WhiteCol` presents coefficients from the comparison of W to M. The second panel, labeled `Prof`, holds the comparison of P to M. Accordingly, the top panel should be compared with the coefficients from the binary logit for W and M (outcome variable `wc_man`) listed above. For example, the coefficient for the comparison of W to M from `mlogit` is $\widehat{\beta}_{1,W|M} = .3000735$ with $z = 3.57$, whereas the `logit` estimate is $\widehat{\beta}_{1,W|M} = .3418255$ with $z = 3.66$. Overall, the estimates from the binary model are close to those from the MNLM but not exactly the same.

Although theoretically $\beta_{1,P|M} - \beta_{1,W|M} = \beta_{1,P|W}$, the estimates from the *binary* logits are $\widehat{\beta}_{1,P|M} - \widehat{\beta}_{1,W|M} = .7184599 - .3418255 = .3766344$, which does not equal the binary logit estimate $\widehat{\beta}_{1,P|W} = .3735466$. A series of binary logits using `logit` does *not* impose the constraints among coefficients that are implicit in the definition of the model. When fitting the model with `mlogit`, the constraints are imposed. Indeed, the output from `mlogit` presents only two of the three comparisons from our example, namely, W versus M and P versus M. The remaining comparison, W versus P, is the difference between the two sets of estimated coefficients. Details on using `listcoef` to automatically compute the remaining comparisons are given below.

6.1.1 Formal statement of the model

Formally, the MNLM can be written as

$$\ln \Omega_{m|b}(\mathbf{x}) = \ln \frac{\Pr(y = m \mid \mathbf{x})}{\Pr(y = b \mid \mathbf{x})} = \mathbf{x}\boldsymbol{\beta}_{m|b} \quad \text{for } m = 1 \text{ to } J$$

where b is the base category, which is also referred to as the comparison group. As $\ln \Omega_{b|b}(\mathbf{x}) = \ln 1 = 0$, it must hold that $\boldsymbol{\beta}_{b|b} = 0$. That is, the log odds of an outcome compared with itself are always 0, and thus the effects of any independent variables must also be 0. These J equations can be solved to compute the predicted probabilities:

$$\Pr(y = m \mid \mathbf{x}) = \frac{\exp\left(\mathbf{x}\boldsymbol{\beta}_{m|b}\right)}{\sum_{j=1}^{J} \exp\left(\mathbf{x}\boldsymbol{\beta}_{j|b}\right)}$$

Although the predicted probability will be the same regardless of the base outcome, b, changing the base outcome can be confusing since the resulting output from `mlogit` *appears* to be quite different. Suppose that you have three outcomes and fit the model with alternative 1 as the base category. Your probability equations would be

$$\Pr(y = m \mid \mathbf{x}) = \frac{\exp\left(\mathbf{x}\boldsymbol{\beta}_{m|1}\right)}{\sum_{j=1}^{J} \exp\left(\mathbf{x}\boldsymbol{\beta}_{j|1}\right)}$$

and you would obtain estimates $\widehat{\boldsymbol{\beta}}_{2|1}$ and $\widehat{\boldsymbol{\beta}}_{3|1}$, where $\boldsymbol{\beta}_{1|1} = 0$. If someone else set up the model with base category 2, their equations would be

$$\Pr(y = m \mid \mathbf{x}) = \frac{\exp\left(\mathbf{x}\boldsymbol{\beta}_{m|2}\right)}{\sum_{j=1}^{J} \exp\left(\mathbf{x}\boldsymbol{\beta}_{j|2}\right)}$$

and they would obtain $\widehat{\boldsymbol{\beta}}_{1|2}$ and $\widehat{\boldsymbol{\beta}}_{3|2}$, where $\boldsymbol{\beta}_{2|2} = 0$. Although the estimated parameters are different, they are only different *parameterizations* that provide the same predicted probabilities. The confusion arises only if you are not clear about which parameterization you are using. Unfortunately, some software packages—but *not* Stata—make it hard to tell which set of parameters is being estimated. We return to this issue when we discuss how Stata's `mlogit` parameterizes the model in the next section.

6.2 Estimation using mlogit

The multinomial logit model is fitted with the following command and its basic options:

`mlogit` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] [, `noconstant`
`baseoutcome(#)` `constraints(`*clist*`)` `robust` `cluster(`*varname*`)` `level(#)`
`rrr` `nolog`]

In our experience, the model converges quickly, even when there are many outcome categories and independent variables.

Variable lists

depvar is the dependent variable. The actual values taken on by the dependent variable are irrelevant. For example, if you had three outcomes, you could use the values 1, 2, and 3 or −1, 0, and 999. Up to 50 outcomes are allowed in Stata/SE and Intercooled Stata, and 20 outcomes are allowed in Small Stata.

indepvars is a list of independent variables. If *indepvars* is not included, Stata fits a model with only constants.

Specifying the estimation sample

if and in qualifiers can be used to restrict the estimation sample. For example, if you want to fit the model with only white respondents, use the command `mlogit occ ed exper if white==1`.

Listwise deletion Stata excludes cases in which there are missing values for any of the variables. Accordingly, if two models are fitted using the same dataset but have different sets of independent variables, it is possible to have different samples. We recommend that you use `mark` and `markout` (discussed in chapter 3) to explicitly remove cases with missing data.

Weights

`mlogit` can be used with `fweights`, `pweights`, and `iweights`. In chapter 3, we provide a brief discussion of the different types of weights and how weights are specified in Stata's syntax.

Options

`noconstant` excludes the constant terms from the model.

`baseoutcome(#)` specifies the value of *depvar* that is the base category (i.e., reference group) for the coefficients that are listed. This determines how the model is parameterized. If the `baseoutcome()` option is not specified, the most frequent outcome in the estimation sample is chosen as the base. The base category is always reported immediately below the estimates; for example, `Outcome occ3==Manual is the base outcome`.

`constraints(`*clist*`)` specifies the linear constraints to be applied during estimation. The default is to perform unconstrained estimation. Constraints are defined with the `constraint` command. This option is illustrated in section 6.3.3 when we discuss an LR test for combining outcome categories.

`robust` indicates that robust variance estimates are to be used. When `cluster()` is specified, robust standard errors are automatically used. See chapter 3 for more details.

`cluster(`*varname*`)` specifies that the observations be independent across the groups specified by unique values of *varname* but not necessarily independent within the groups. See chapter 3 for more details.

`level(#)` specifies the level of the confidence interval for estimated parameters. By default, Stata uses 95% intervals. You can also change the default level to, say, a 90% interval, with the command `set level 90`.

`rrr` reports the estimated coefficients transformed to relative risk ratios, defined as $\exp(b)$ rather than b, along with standard errors and confidence intervals for these ratios.

`nolog` suppresses the iteration history.

6.2.1 Example of occupational attainment

The 1982 General Social Survey asked respondents their occupation, which we recoded into five broad categories: menial jobs (M), blue collar jobs (B), craft jobs (C), white collar jobs (W), and professional jobs (P). Three independent variables are considered: `white` indicating the race of the respondent, `ed` measuring years of education, and `exper` measuring years of work experience.

```
. summarize white ed exper
```

Variable	Obs	Mean	Std. Dev.	Min	Max
white	337	.9169139	.2764227	0	1
ed	337	13.09496	2.946427	3	20
exper	337	20.50148	13.95936	2	66

The distribution among outcome categories is

```
. tab occ
```

Occupation	Freq.	Percent	Cum.
Menial	31	9.20	9.20
BlueCol	69	20.47	29.67
Craft	84	24.93	54.60
WhiteCol	41	12.17	66.77
Prof	112	33.23	100.00
Total	337	100.00	

Using these variables, the following MNLM was fitted:

$$
\begin{aligned}
\ln \Omega_{M|P}(\mathbf{x}_i) &= \beta_{0,M|P} + \beta_{1,M|P}\texttt{white} + \beta_{2,M|P}\texttt{ed} + \beta_{3,M|P}\texttt{exper} \\
\ln \Omega_{B|P}(\mathbf{x}_i) &= \beta_{0,B|P} + \beta_{1,B|P}\texttt{white} + \beta_{2,B|P}\texttt{ed} + \beta_{3,B|P}\texttt{exper} \\
\ln \Omega_{C|P}(\mathbf{x}_i) &= \beta_{0,C|P} + \beta_{1,C|P}\texttt{white} + \beta_{2,C|P}\texttt{ed} + \beta_{3,C|P}\texttt{exper} \\
\ln \Omega_{W|P}(\mathbf{x}_i) &= \beta_{0,W|P} + \beta_{1,W|P}\texttt{white} + \beta_{2,W|P}\texttt{ed} + \beta_{3,W|P}\texttt{exper}
\end{aligned}
$$

where we specify the fifth outcome P as the base category:


```
. mlogit occ white ed exper, baseoutcome(5) nolog
Multinomial logistic regression                   Number of obs   =        337
                                                  LR chi2(12)     =     166.09
                                                  Prob > chi2     =     0.0000
Log likelihood = -426.80048                       Pseudo R2       =     0.1629
```

occ	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Menial						
white	-1.774306	.7550543	-2.35	0.019	-3.254186	-.2944273
ed	-.7788519	.1146293	-6.79	0.000	-1.003521	-.5541826
exper	-.0356509	.018037	-1.98	0.048	-.0710028	-.000299
_cons	11.51833	1.849356	6.23	0.000	7.893659	15.143
BlueCol						
white	-.5378027	.7996033	-0.67	0.501	-2.104996	1.029391
ed	-.8782767	.1005446	-8.74	0.000	-1.07534	-.6812128
exper	-.0309296	.0144086	-2.15	0.032	-.05917	-.0026893
_cons	12.25956	1.668144	7.35	0.000	8.990061	15.52907
Craft						
white	-1.301963	.647416	-2.01	0.044	-2.570875	-.0330509
ed	-.6850365	.0892996	-7.67	0.000	-.8600605	-.5100126
exper	-.0079671	.0127055	-0.63	0.531	-.0328693	.0169351
_cons	10.42698	1.517943	6.87	0.000	7.451864	13.40209
WhiteCol						
white	-.2029212	.8693072	-0.23	0.815	-1.906732	1.50089
ed	-.4256943	.0922192	-4.62	0.000	-.6064407	-.2449479
exper	-.001055	.0143582	-0.07	0.941	-.0291967	.0270866
_cons	5.279722	1.684006	3.14	0.002	1.979132	8.580313

```
(occ==Prof is the base outcome)
```

Methods of testing coefficients and interpretation of the estimates will be considered after we discuss the effects of using different base categories.

6.2.2 Using different base categories

By default, `mlogit` sets the base category to the alternative with the most observations. Or, as illustrated in the last example, you can select the base category with `baseoutcome()`. `mlogit` then reports coefficients for the effect of each independent variable on each category relative to the base category. However, you should also examine the effects on other pairs of outcome categories. For example, you might be interested in how race affects the allocation of workers between `Craft` and `BlueCol` (e.g., $\beta_{1,B|C}$), which was not estimated in the output listed above. Although this coefficient can be estimated by rerunning `mlogit` with a different base category (e.g., `mlogit occ white ed exper, baseoutcome(3)`), it is easier to use `listcoef`, which presents estimates for *all* combinations of outcome categories. Because `listcoef` can generate much output, we show two options that limit which coefficients are listed. First, you can include a list of variables, and only coefficients for those variables will be listed. For example,

```
. listcoef white, help
mlogit (N=337): Factor Change in the Odds of occ
Variable: white (sd=.27642268)
```

Odds comparing Alternative 1 to Alternative 2	b	z	P>|z|	e^b	e^bStdX
Menial -BlueCol	-1.23650	-1.707	0.088	0.2904	0.7105
Menial -Craft	-0.47234	-0.782	0.434	0.6235	0.8776
Menial -WhiteCol	-1.57139	-1.741	0.082	0.2078	0.6477
Menial -Prof	-1.77431	-2.350	0.019	0.1696	0.6123
BlueCol -Menial	1.23650	1.707	0.088	3.4436	1.4075
BlueCol -Craft	0.76416	1.208	0.227	2.1472	1.2352
BlueCol -WhiteCol	-0.33488	-0.359	0.720	0.7154	0.9116
BlueCol -Prof	-0.53780	-0.673	0.501	0.5840	0.8619
Craft -Menial	0.47234	0.782	0.434	1.6037	1.1395
Craft -BlueCol	-0.76416	-1.208	0.227	0.4657	0.8096
Craft -WhiteCol	-1.09904	-1.343	0.179	0.3332	0.7380
Craft -Prof	-1.30196	-2.011	0.044	0.2720	0.6978
WhiteCol-Menial	1.57139	1.741	0.082	4.8133	1.5440
WhiteCol-BlueCol	0.33488	0.359	0.720	1.3978	1.0970
WhiteCol-Craft	1.09904	1.343	0.179	3.0013	1.3550
WhiteCol-Prof	-0.20292	-0.233	0.815	0.8163	0.9455
Prof -Menial	1.77431	2.350	0.019	5.8962	1.6331
Prof -BlueCol	0.53780	0.673	0.501	1.7122	1.1603
Prof -Craft	1.30196	2.011	0.044	3.6765	1.4332
Prof -WhiteCol	0.20292	0.233	0.815	1.2250	1.0577

```
        b = raw coefficient
        z = z-score for test of b=0
    P>|z| = p-value for z-test
      e^b = exp(b) = factor change in odds for unit increase in X
  e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
```

Or, you can limit the output to those coefficients that are significant at a given level using the pvalue(#) option, which specifies that only coefficients significant at the # significance level or smaller will be printed. For example,

(Continued on next page)


```
. listcoef, pvalue(.05)
mlogit (N=337): Factor Change in the Odds of occ when P>|z| < 0.05
Variable: white (sd=.27642268)
```

Odds comparing Alternative 1 to Alternative 2	b	z	P>|z|	e^b	e^bStdX
Menial -Prof	-1.77431	-2.350	0.019	0.1696	0.6123
Craft -Prof	-1.30196	-2.011	0.044	0.2720	0.6978
Prof -Menial	1.77431	2.350	0.019	5.8962	1.6331
Prof -Craft	1.30196	2.011	0.044	3.6765	1.4332

```
Variable: ed (sd=2.9464271)
```

Odds comparing Alternative 1 to Alternative 2	b	z	P>|z|	e^b	e^bStdX
Menial -WhiteCol	-0.35316	-3.011	0.003	0.7025	0.3533
Menial -Prof	-0.77885	-6.795	0.000	0.4589	0.1008
BlueCol -Craft	-0.19324	-2.494	0.013	0.8243	0.5659
BlueCol -WhiteCol	-0.45258	-4.425	0.000	0.6360	0.2636
BlueCol -Prof	-0.87828	-8.735	0.000	0.4155	0.0752
Craft -BlueCol	0.19324	2.494	0.013	1.2132	1.7671
Craft -WhiteCol	-0.25934	-2.773	0.006	0.7716	0.4657
Craft -Prof	-0.68504	-7.671	0.000	0.5041	0.1329
WhiteCol-Menial	0.35316	3.011	0.003	1.4236	2.8308
WhiteCol-BlueCol	0.45258	4.425	0.000	1.5724	3.7943
WhiteCol-Craft	0.25934	2.773	0.006	1.2961	2.1471
WhiteCol-Prof	-0.42569	-4.616	0.000	0.6533	0.2853
Prof -Menial	0.77885	6.795	0.000	2.1790	9.9228
Prof -BlueCol	0.87828	8.735	0.000	2.4067	13.3002
Prof -Craft	0.68504	7.671	0.000	1.9838	7.5264
Prof -WhiteCol	0.42569	4.616	0.000	1.5307	3.5053

```
Variable: exper (sd=13.959364)
```

Odds comparing Alternative 1 to Alternative 2	b	z	P>|z|	e^b	e^bStdX
Menial -Prof	-0.03565	-1.977	0.048	0.9650	0.6079
BlueCol -Prof	-0.03093	-2.147	0.032	0.9695	0.6494
Prof -Menial	0.03565	1.977	0.048	1.0363	1.6449
Prof -BlueCol	0.03093	2.147	0.032	1.0314	1.5400

If you do not need to see the comparisons between all pairs of alternatives, you can limit the output with the `gt` or `lt` options of `listcoef`. By default, `listcoef` lists comparisons in both directions. For example, it will show you the effect on the odds of alternative 1 versus alternative 2 and the effect on the odds of 2 versus 1. The `gt` option limits comparisons to those in which the first alternative is greater than the second; `lt` shows comparisons when the first alternative is less than the second. For example,

```
. listcoef ed, pvalue(.05) gt nolabel
mlogit (N=337): Factor Change in the Odds of occ when P>|z| < 0.05
Variable: ed (sd=2.9464271)
Odds comparing
Alternative 1
to Alternative 2        b         z     P>|z|     e^b    e^bStdX

3      -2         0.19324    2.494    0.013   1.2132    1.7671
4      -1         0.35316    3.011    0.003   1.4236    2.8308
4      -2         0.45258    4.425    0.000   1.5724    3.7943
4      -3         0.25934    2.773    0.006   1.2961    2.1471
5      -1         0.77885    6.795    0.000   2.1790    9.9228
5      -2         0.87828    8.735    0.000   2.4067   13.3002
5      -3         0.68504    7.671    0.000   1.9838    7.5264
5      -4         0.42569    4.616    0.000   1.5307    3.5053
```

We used the nolabel option to show the category values of the two alternatives rather than their value labels, and the pvalue(.05) option limits the coefficients that are printed to those that are significant at the .05 level.

6.2.3 Predicting perfectly

mlogit handles perfect prediction somewhat differently than the estimations commands for binary and ordinal models that we have discussed. logit and probit automatically remove the observations that imply perfect prediction and compute estimates accordingly. ologit and oprobit keep these observations in the model, fit the z for the problem variable as 0, and provide an incorrect LR chi-squared but also warn that a given number of observations are completely determined. You should delete these observations and refit the model. mlogit is just like ologit and oprobit, except that *~~you do not receive a warning message~~*. ~~You will see, however, that all coefficients associated~~ with the variable causing the problem have $z = 0$ (and $p > |z| \doteq 1$). You should refit the model, excluding the problem variable and deleting the observations that imply the perfect predictions. Using the tabulate command to generate a cross-tabulation of the problem variable and the dependent variable should reveal the combination that results in perfect prediction.

6.3 Hypothesis testing of coefficients

In the MNLM, you can test individual coefficients with the reported z-statistics, with a Wald test using test, or with an LR test using lrtest. As the methods of testing one coefficient that were discussed in chapters 4 and 5 still apply fully, they are not considered further here. However, in the MNLM there are new reasons for testing groups of coefficients. First, testing that a variable has no effect requires a test that $J-1$ coefficients are simultaneously equal to zero. Second, testing whether the independent variables as a group differentiate between two alternatives requires a test of K coefficients. This section focuses on these two kinds of tests.

Caution regarding specification searches Given the difficulties of interpretation that are associated with the MNLM, it is tempting to search for a more parsimonious model by excluding variables or combining outcome categories based on a sequence of tests. Such a search requires great care. First, these tests involve multiple coefficients. Although the overall test might indicate that *as a group* the coefficients are not significantly different from zero, an *individual* coefficient can still be substantively and statistically significant. Accordingly, you should examine the individual coefficients involved in each test before deciding to revise your model. Second, as with all searches that use repeated, sequential tests, there is a danger of overfitting the data. When models are constructed based on prior testing using the same data, significance levels should be used only as rough guidelines.

6.3.1 mlogtest for tests of the MNLM

Although the tests in this section can be computed using test or lrtest, in practice this is tedious. The mlogtest command (Freese and Long 2000) makes the computation of these tests easy. The syntax is

mlogtest [*varlist*] [, all lr wald combine lrcomb
set(*varlist*[\ *varlist*[\...]]) iia hausman smhsiao detail base]

varlist indicates that the variables for which tests of significance should be computed. If no *varlist* is given, tests are run for all independent variables.

Options

lr requests a likelihood-ratio (LR) test for each variable in *varlist*. If *varlist* is not specified, tests for all variables are computed.

wald requests a Wald test for each variable in *varlist*. If *varlist* is not specified, tests for all variables are computed.

combine requests Wald tests of whether dependent categories can be combined.

lrcomb requests LR tests of whether dependent categories can be combined. These tests use constrained estimation and overwrite constraint #999 if it is already defined.

set(*varlist*[\ *varlist*[\...]]) specifies that a set of variables is to be considered together for the LR test or Wald test. \ is used to specify multiple sets of variables. For example, mlogtest, lr set(age age2 \ iscat1 iscat2) computes one LR test for the hypothesis that the effects of age and age2 are jointly 0 and a second LR test that the effects of iscat1 and iscat2 are jointly 0.

Other options for mlogtest are discussed later in the chapter.

6.3.2 Testing the effects of the independent variables

With J dependent categories, there are $J-1$ nonredundant coefficients associated with each independent variable x_k. For example, in our logit on occupation, there are four coefficients associated with ed: $\beta_{2,M|P}$, $\beta_{2,B|P}$, $\beta_{2,C|P}$, and $\beta_{2,W|P}$. The hypothesis that x_k does not affect the dependent variable can be written as

$$H_0\colon \beta_{k,1|b} = \cdots = \beta_{k,J|b} = 0$$

where b is the base category. Because $\beta_{k,b|b}$ is necessarily 0, the hypothesis imposes constraints on $J-1$ parameters. This hypothesis can be tested with either a Wald or an LR test.

A likelihood-ratio test

The LR test involves (1) fitting the full model, including all the variables, resulting in the likelihood-ratio statistic LR^2_F; (2) fitting the restricted model that excludes variable x_k, resulting in LR^2_R; and (3) computing the difference $LR^2_{R\text{vs}F} = LR^2_F - LR^2_R$, which is distributed as chi-squared with $J-1$ degrees of freedom if the null hypothesis is true. This can be done using `lrtest`:

```
. use http://www.stata-press.com/data/lf2/nomocc2, clear
(1982 General Social Survey)
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. estimates store fmodel
. mlogit occ ed exper, baseoutcome(5) nolog
  (output omitted)
. estimates store nmodel_white
. lrtest fmodel nmodel_white
Likelihood-ratio test                                 LR chi2(4)  =      8.10
(Assumption: nmodel_white nested in fmodel)           Prob > chi2 =    0.0881
. mlogit occ white exper, baseoutcome(5) nolog
  (and so on)
```

Although using `lrtest` is straightforward, the command `mlogtest, lr` is even simpler because it automatically computes the tests for all variables by making repeated calls to `lrtest`:

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. mlogtest, lr
**** Likelihood-ratio tests for independent variables (N=337)
 Ho: All coefficients associated with given variable(s) are 0.

         occ |      chi2   df   P>chi2
-------------+-------------------------
       white |     8.095    4    0.088
          ed |   156.937    4    0.000
       exper |     8.561    4    0.073
---------------------------------------
```


The results of the LR test, regardless of how they are computed, can be interpreted as follows:

> The effect of race on occupation is significant at the .10 level but not at the .05 level ($X^2 = 8.10$, $df = 4$, $p = .09$). The effect of education is significant at the .01 level ($X^2 = 156.94$, $df = 4$, $p < .01$).

Or, it can be stated more formally:

> The hypothesis that all the coefficients associated with education are simultaneously equal to 0 can be rejected at the .01 level ($X^2 = 156.94$, $df = 4$, $p < .01$).

A Wald test

Although the LR test is generally considered superior, its computational costs can be prohibitive if the model is complex or the sample is very large. K Wald tests can also be computed using `test` without fitting additional models. For example,

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. test white
 ( 1)  [Menial]white = 0
 ( 2)  [BlueCol]white = 0
 ( 3)  [Craft]white = 0
 ( 4)  [WhiteCol]white = 0
           chi2(  4) =    8.15
         Prob > chi2 =    0.0863
. test ed
 ( 1)  [Menial]ed = 0
 ( 2)  [BlueCol]ed = 0
 ( 3)  [Craft]ed = 0
 ( 4)  [WhiteCol]ed = 0
           chi2(  4) =   84.97
         Prob > chi2 =    0.0000
. test exper
 ( 1)  [Menial]exper = 0
 ( 2)  [BlueCol]exper = 0
 ( 3)  [Craft]exper = 0
 ( 4)  [WhiteCol]exper = 0
           chi2(  4) =    7.99
         Prob > chi2 =    0.0918
```

The output from `test` makes explicit which coefficients are being tested. Here we see the way in which Stata labels parameters in models with multiple equations. For example, `[Menial]white` is the coefficient for the effect of `white` in the equation comparing the outcome `Menial` with the base category `Prof`; `[BlueCol]white` is the coefficient for the effect of `white` in the equation comparing the outcome `BlueCol` with the base category `Prof`.

As with the LR test, mlogtest, wald automates this process:

```
. mlogtest, wald
**** Wald tests for independent variables (N=337)
 Ho: All coefficients associated with given variable(s) are 0.
         occ |     chi2   df   P>chi2
-------------+-------------------------
       white |    8.149    4    0.086
          ed |   84.968    4    0.000
       exper |    7.995    4    0.092
---------------------------------------
```

These tests can be interpreted in the same way as shown for the LR test above.

Testing multiple independent variables

The logic of the Wald or LR tests can be extended to test that the effects of two or more independent variables are simultaneously zero. For example, the hypothesis to test that x_k and x_ℓ have no effect is

$$H_0\colon \beta_{k,1|b} = \cdots = \beta_{k,J|b} = \beta_{\ell,1|b} = \cdots = \beta_{\ell,J|b} = 0$$

The set(*varlist*[\ *varlist*[\...]]) option in mlogtest specifies which variables are to be simultaneously tested. For example, to test the hypothesis that the effects of ed and exper are simultaneously equal to 0, we could use lrtest as follows:

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. estimates store fmodel
. mlogit occ white, baseoutcome(5) nolog
  (output omitted)
. estimates store nmodel
. lrtest fmodel nmodel
Likelihood-ratio test                                 LR chi2(8)  =    160.77
(Assumption: nmodel nested in fmodel)                 Prob > chi2 =    0.0000
```


or, using mlogtest,

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. mlogtest, lr set(ed exper)
**** Likelihood-ratio tests for independent variables (N=337)
 Ho: All coefficients associated with given variable(s) are 0.
         occ |     chi2   df   P>chi2
-------------+-------------------------
       white |    8.095    4    0.088
          ed |  156.937    4    0.000
       exper |    8.561    4    0.073
-------------+-------------------------
      set_1: |  160.773    8    0.000
          ed |
       exper |
---------------------------------------
```

6.3.3 Tests for combining alternatives

If none of the independent variables significantly affect the odds of alternative m versus alternative n, we say that m and n are *indistinguishable* with respect to the variables in the model (Anderson 1984). Alternatives m and n's being indistinguishable corresponds to the hypothesis that

$$H_0\colon \beta_{1,m|n} = \cdots \beta_{K,m|n} = 0$$

which can be tested with either a Wald or an LR test. In our experience, the two tests provide similar results. If alternatives are indistinguishable with respect to the variables in the model, then you can obtain more efficient estimates by combining them. To test whether alternatives are indistinguishable, you can use mlogtest.

A Wald test for combining alternatives

The command mlogtest, combine computes Wald tests of the null hypothesis that two alternatives can be combined for all pairs of alternatives. For example,

(*Continued on next page*)

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. mlogtest, combine
**** Wald tests for combining alternatives (N=337)
 Ho: All coefficients except intercepts associated with a given pair
     of alternatives are 0 (i.e., alternatives can be combined).
Alternatives tested|     chi2   df   P>chi2
-------------------+------------------------
  Menial- BlueCol  |    3.994    3    0.262
  Menial-   Craft  |    3.203    3    0.361
  Menial-WhiteCol  |   11.951    3    0.008
  Menial-    Prof  |   48.190    3    0.000
 BlueCol-   Craft  |    8.441    3    0.038
 BlueCol-WhiteCol  |   20.055    3    0.000
 BlueCol-    Prof  |   76.393    3    0.000
   Craft-WhiteCol  |    8.892    3    0.031
   Craft-    Prof  |   60.583    3    0.000
WhiteCol-    Prof  |   22.203    3    0.000
--------------------------------------------
```

For example, we can reject the hypothesis that categories Menial and Prof are indistinguishable, whereas we cannot reject that Menial and BlueCol are indistinguishable.

Using test [category]*

The mlogtest command computes the tests for combining categories with the test command. For example, to test that Menial is indistinguishable from the base category Prof, type

```
. test [Menial]
 ( 1)  [Menial]white = 0
 ( 2)  [Menial]ed = 0
 ( 3)  [Menial]exper = 0
           chi2(  3) =   48.19
         Prob > chi2 =    0.0000
```

which matches the results from mlogtest in row Menial-Prof. [*outcome*] in test is used to indicate which equation is being referenced in multiple equation commands. mlogit is a multiple equation command because it is in effect estimating $J-1$ binary logit equations.

The test is more complicated when neither outcome is the base category. For example, to test that m and n are indistinguishable when the base category b is neither m nor n, the hypothesis you want to test is

$$H_0\colon\ (\beta_{1,m|b}-\beta_{1,n|b}) = \cdots = (\beta_{K,m|b}-\beta_{K,n|b}) = 0$$

That is, you want to test the difference between two sets of coefficients. This can be done with test [*outcome1*=*outcome2*]. For example, to test if Menial and Craft can be combined, type


```
. test [Menial=Craft]
 ( 1)  [Menial]white - [Craft]white = 0
 ( 2)  [Menial]ed - [Craft]ed = 0
 ( 3)  [Menial]exper - [Craft]exper = 0
           chi2(  3) =    3.20
         Prob > chi2 =    0.3614
```

Again the results are identical to those from mlogtest.

An LR test for combining alternatives

An LR test of combining m and n can be computed by first fitting the full model with no constraints, with the resulting LR statistic LR^2_F. Then we fit a restricted model M_R in which outcome m is used as the base category and all the coefficients except the constant in the equation for outcome n are constrained to 0, with the resulting test statistic LR^2_R. The test statistic is the difference $LR^2_{R\text{vs}F} = LR^2_F - LR^2_R$, which is distributed as chi-squared with K degrees of freedom. The command mlogtest, lrcomb computes $J \times (J-1)$ tests for all pairs of outcome categories. For example,

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. mlogtest, lrcomb
**** LR tests for combining alternatives (N=337)
 Ho: All coefficients except intercepts associated with a given pair
     of alternatives are 0 (i.e., alternatives can be collapsed).
Alternatives tested|     chi2   df   P>chi2
-------------------+------------------------
  Menial- BlueCol  |    4.095    3    0.251
  Menial-   Craft  |    3.376    3    0.337
  Menial-WhiteCol  |   13.223    3    0.004
  Menial-    Prof  |   64.607    3    0.000
 BlueCol-   Craft  |    9.176    3    0.027
 BlueCol-WhiteCol  |   22.803    3    0.000
 BlueCol-    Prof  |  125.699    3    0.000
   Craft-WhiteCol  |    9.992    3    0.019
   Craft-    Prof  |   95.889    3    0.000
WhiteCol-    Prof  |   26.736    3    0.000
--------------------------------------------
```

Using constraint with lrtest*

The command mlogtest, lrcomb computes the test by using the powerful constraint command. To show this, we use the test comparing Menial and BlueCol reported by mlogtest, lrcomb above. First, we fit the full model and save the results of lrtest:

```
. mlogit occ white ed exper, nolog
  (output omitted)
. estimates store fmodel
```

Second, we define a constraint using the command

```
. constraint define 999 [Menial]
```

This defines constraint 999, where the number is arbitrary. The expression `[Menial]` indicates that all the coefficients except the constant from the `Menial` equation should be constrained to 0. Third, we refit the model with this constraint. The base category must be `BlueCol`, so that the coefficients indicated by `[Menial]` are comparisons of `BlueCol` and `Menial`:

```
. mlogit occ exper ed white, base(2) constraint(999) nolog
Multinomial logistic regression                   Number of obs   =        337
                                                  LR chi2(9)      =     161.99
                                                  Prob > chi2     =     0.0000
Log likelihood = -428.84791                       Pseudo R2       =     0.1589
 ( 1)  [Menial]exper = 0
 ( 2)  [Menial]ed = 0
 ( 3)  [Menial]white = 0
```

occ	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Menial						
exper	(dropped)					
ed	(dropped)					
white	(dropped)					
_cons	-.8001193	.2162194	-3.70	0.000	-1.223901	-.3763371
Craft						
exper	.0242824	.0113959	2.13	0.033	.0019469	.0466179
ed	.1599345	.0693853	2.31	0.021	.0239418	.2959273
white	-.2381783	.4978563	-0.48	0.632	-1.213959	.7376021
_cons	-1.969087	1.054935	-1.87	0.062	-4.036721	.098547
WhiteCol						
exper	.0312007	.0143598	2.17	0.030	.0030561	.0593454
ed	.4195709	.0958978	4.38	0.000	.2316147	.607527
white	.8829927	.843371	1.05	0.295	-.7699841	2.535969
_cons	-7.140306	1.623401	-4.40	0.000	-10.32211	-3.958498
Prof						
exper	.032303	.0133779	2.41	0.016	.0060827	.0585233
ed	.8445092	.093709	9.01	0.000	.6608429	1.028176
white	1.097459	.6877939	1.60	0.111	-.2505923	2.44551
_cons	-12.42143	1.569897	-7.91	0.000	-15.49837	-9.344489

```
(occ==BlueCol is the base outcome)
```

`mlogit` requires the option `constraint(999)` to indicate that estimation should impose this constraint. The output clearly indicates which constraints have been imposed. Finally, we use `lrtest` to compute the test:

```
. estimates store nmodel
. lrtest fmodel nmodel
Likelihood-ratio test                                 LR chi2(3)  =      4.09
(Assumption: nmodel nested in fmodel)                 Prob > chi2 =    0.2514
```


6.4 Independence of irrelevant alternatives

Both the MNLM and the conditional logit model (discussed below) make the assumption known as the *independence of irrelevant alternatives* (IIA). Here we describe the assumption in terms of the MNLM. In this model,

$$\frac{\Pr(y=m \mid \mathbf{x})}{\Pr(y=n \mid \mathbf{x})} = \exp\left\{\mathbf{x}\left(\boldsymbol{\beta}_{m|b} - \boldsymbol{\beta}_{n|b}\right)\right\}$$

where the odds do not depend on other alternatives that are available. In this sense, these alternatives are "irrelevant". What this means is that adding or deleting alternatives does not affect the odds among the remaining alternatives. This point is often made with the red bus–blue bus example. Suppose that you have the choice of a red bus or a car to get to work and that the odds of taking a red bus compared with those of taking a car are 1:1. IIA implies that the odds will remain 1:1 between these two alternatives, even if a new *blue* bus company comes to town that is identical to the red bus company, except for the color of the bus. Thus the probability of driving a car can be made arbitrarily small by adding enough different colors of buses! More reasonably, we might expect that the odds of a red bus compared with those of a car would be reduced to 1:2 since half of those riding the red bus would be expected to ride the blue bus.

Tests of IIA involve comparing the estimated coefficients from the full model to those from a restricted model that excludes at least one of the alternatives. If the test statistic is significant, the assumption of IIA is rejected indicating that the MNLM is inappropriate. In this section, we consider the two most common tests of IIA: the Hausman–McFadden (HM) test (1984) and the Small–Hsiao (SH) test (1985). For details on other tests, see Fry and Harris (1996, 1998). In a model with J alternatives, there are $J-1$ ways of computing each test. If you remove the first alternative and refit the model, you get the first restricted model. If you remove the second alternative, the second, and so on, for a total of $J-1$ restricted models, each of these restricted models will lead to a different test statistic, as we demonstrate below.

Both the HM and the SH tests are computed by `mlogtest`, and for both tests we compute $J-1$ variations. As many users of `mlogtest` have told us, the HM and SH tests often provide conflicting information on whether IIA has been violated (i.e., some of the tests reject the null hypothesis, whereas others do not). To explore this further, Cheng and Long (2005) ran Monte Carlo experiments to examine the properties of these tests. Their results show that the HM test has poor size properties even with sample sizes of more than 1,000. For some data structures, the SH test has reasonable size properties for samples of 500 or more. But, with other data structures the size properties are extremely poor and do not get better as the sample size increases. Overall, they conclude that these tests are not useful for assessing violations of the IIA property. It appears that the best advice regarding IIA goes back to an early statement by McFadden (1973), who wrote that the multinomial and conditional logit models should be used only in cases where the alternatives "can plausibly be assumed to be distinct and weighted independently in the eyes of each decision maker". Similarly, Amemiya

(1981, 1,517) suggests that the MNLM works well when the alternatives are dissimilar. Care in specifying the model to involve distinct alternatives that are not substitutes for one another seems to be reasonable, albeit unfortunately ambiguous, advice. Nonetheless, we continue to include these tests in `mlogtest`, but we do not encourage their use. As we will show here, these tests can produce contradictory results.

Hausman test of IIA

The Hausman test of IIA involves the following steps:

1. Fit the full model with all J alternatives included, with estimates in $\widehat{\boldsymbol\beta}_F$.
2. Fit a restricted model by eliminating one or more alternatives, with estimates in $\widehat{\boldsymbol\beta}_R$.
3. Let $\widehat{\boldsymbol\beta}_F^*$ be a subset of $\widehat{\boldsymbol\beta}_F$ after eliminating coefficients not fitted in the restricted model. The test statistic is

$$H = \left(\widehat{\boldsymbol\beta}_R - \widehat{\boldsymbol\beta}_F^*\right)' \left\{\widehat{\text{Var}}\left(\widehat{\boldsymbol\beta}_R\right) - \widehat{\text{Var}}\left(\widehat{\boldsymbol\beta}_F^*\right)\right\}^{-1} \left(\widehat{\boldsymbol\beta}_R - \widehat{\boldsymbol\beta}_F^*\right)$$

where H is asymptotically distributed as chi-squared with degrees of freedom equal to the rows in $\widehat{\boldsymbol\beta}_R$ if IIA is true. Significant values of H indicate that the IIA assumption has been violated.

The Hausman test of IIA can be computed with `mlogtest`. Here the results are

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. mlogtest, hausman base
**** Hausman tests of IIA assumption (N=337)
 Ho: Odds(Outcome-J vs Outcome-K) are independent of other alternatives.
```

Omitted	chi2	df	P>chi2	evidence
Menial	7.324	12	0.835	for Ho
BlueCol	0.320	12	1.000	for Ho
Craft	-14.436	12	1.000	for Ho
WhiteCol	-5.541	11	1.000	for Ho
Prof	-0.119	12	1.000	for Ho

Five tests of IIA are reported. The first four correspond to excluding one of the four nonbase categories. The fifth test, in row `Prof`, is computed by refitting the model using the largest remaining outcome as the base category.[1] Although none of the tests reject the H_0 that IIA holds, the results differ considerably, depending on the outcome considered. Further, three of the test statistics are negative, which we find to be very

1. Even though `mlogtest` fits other models to compute various tests, when the command ends it restores the estimates from your original model. Accordingly, other commands that require results from your original `mlogit`, such as `predict` and `prvalue`, will still work correctly.

common. Hausman and McFadden (1984, 1226) note this possibility and conclude that a negative result is evidence that IIA has *not* been violated. A further sense of the variability of the results can be seen by rerunning `mlogit` with a different base category and then running `mlogtest, hausman base`.

Small–Hsiao test of IIA

To compute Small and Hsiao's test, the sample is divided randomly into two subsamples of about equal size. The unrestricted MNLM is fitted on both subsamples, where $\widehat{\boldsymbol\beta}_u^{S_1}$ contains estimates from the unrestricted model on the first subsample and $\widehat{\boldsymbol\beta}_u^{S_2}$ is its counterpart for the second subsample. A weighted average of the coefficients is computed as

$$\widehat{\boldsymbol\beta}_u^{S_1S_2} = \left(\frac{1}{\sqrt{2}}\right)\widehat{\boldsymbol\beta}_u^{S_1} + \left\{1 - \left(\frac{1}{\sqrt{2}}\right)\right\}\widehat{\boldsymbol\beta}_u^{S_2}$$

Next a restricted sample is created from the second subsample by eliminating all cases with a chosen value of the dependent variable. The MNLM is fitted using the restricted sample, yielding the estimates $\widehat{\boldsymbol\beta}_r^{S_2}$ and the likelihood $L(\widehat{\boldsymbol\beta}_r^{S_2})$. The Small–Hsiao statistic is

$$SH = -2\left\{L(\widehat{\boldsymbol\beta}_u^{S_1S_2}) - L(\widehat{\boldsymbol\beta}_r^{S_2})\right\}$$

which is asymptotically distributed as a chi-squared with the degrees of freedom equal to the number of coefficients that are fitted both in the full model and the restricted model.

To compute the Small–Hsiao test, you use the command `mlogtest, smhsiao` (our program uses code from `smhsiao` by Nick Winter, available at the SSC-IDEAS archive). For example,

```
. mlogtest, smhsiao
**** Small-Hsiao tests of IIA assumption (N=337)
 Ho: Odds(Outcome-J vs Outcome-K) are independent of other alternatives.
```

Omitted	lnL(full)	lnL(omit)	chi2	df	P>chi2	evidence
Menial	-182.140	-169.907	24.466	12	0.018	against Ho
BlueCol	-148.711	-140.054	17.315	12	0.138	for Ho
Craft	-131.801	-119.286	25.030	12	0.015	against Ho
WhiteCol	-161.436	-148.550	25.772	12	0.012	against Ho

In three variations of the SH test, we reject the null, whereas the HM test accepted the null in all cases.

Because the Small–Hsiao test requires randomly dividing the data into subsamples, the results will differ with successive calls of the command, as the sample will be divided differently. To obtain test results that can be replicated, you must explicitly set the seed used by the random-number generator. For example,

```
. set seed 8675309
. mlogtest, smhsiao
**** Small-Hsiao tests of IIA assumption (N=337)
 Ho: Odds(Outcome-J vs Outcome-K) are independent of other alternatives.
```

Omitted	lnL(full)	lnL(omit)	chi2	df	P>chi2	evidence
Menial	-169.785	-161.523	16.523	12	0.168	for Ho
BlueCol	-131.900	-125.871	12.058	12	0.441	for Ho
Craft	-136.934	-129.905	14.058	12	0.297	for Ho
WhiteCol	-155.364	-150.239	10.250	12	0.594	for Ho

Using a new seed, we accept the null in each case, illustrating a common problem when using the SH test—you can get quite different results depending on how the sample is randomly divided.

Advanced: setting the random seed The random numbers that divide the sample for the Small–Hsiao test are based on Stata's `uniform()` function, which uses a pseudorandom number generator. This generator creates a sequence of numbers based on a seed number. Although these numbers appear to be random, the same sequence will be generated each time you start with the same seed number. In this sense (and some others), these numbers are pseudorandom rather than random. If you specify the seed with `set seed #`, you ensure that you can replicate your results later. See the *Data Management Reference Manual* for more details.

6.5 Measures of fit

As with the binary and ordinal models, scalar measures of fit for the MNLM model can be computed with the SPost command `fitstat`. The same caveats against overstating the importance of these scalar measures apply here as to the other models we consider (see also chapter 3). To examine the fit of individual observations, you can estimate the series of binary logits implied by the multinomial logit model and use the established methods of examining the fit of observations to binary logit estimates. This is the same approach that was recommended in chapter 5 for ordinal models.

6.6 Interpretation

Although the MNLM is a mathematically simple extension of the binary model, interpretation is made difficult by the many possible comparisons. Even in our simple example with five outcomes, we have many possible comparisons: $M|P$, $B|P$, $C|P$, $W|P$, $M|W$, $B|W$, $C|W$, $M|C$, $B|C$, and $M|B$. It is tedious to write all the comparisons, let alone to interpret each of them for each of the independent variables. Thus the key to interpretation is to avoid being overwhelmed by the many comparisons. Most of the methods

we propose are similar to those for ordinal outcomes, and accordingly, these are treated briefly. However, methods of plotting discrete changes and factor changes are new, so these are considered in greater detail.

6.6.1 Predicted probabilities

Predicted probabilities can be computed with the formula

$$\widehat{\Pr}(y=m \mid \mathbf{x}) = \frac{\exp\left(\mathbf{x}\widehat{\boldsymbol{\beta}}_{m|J}\right)}{\sum_{j=1}^{J}\exp\left(\mathbf{x}\widehat{\boldsymbol{\beta}}_{j|J}\right)}$$

where $\mathbf{x}$ can contain values from individuals in the sample or hypothetical values. The most basic command for computing probabilities is `predict`, but we also illustrate a series of SPost commands that compute predicted probabilities in useful ways.

6.6.2 Predicted probabilities with predict

After fitting the model with `mlogit`, the predicted probabilities within the sample can be calculated with the command

`predict` *newvar1* [*newvar2* ...[*newvarJ*]] [*if*] [*in*]

where you must provide one new variable name for each of the J categories of the dependent variable, ordered from the lowest to highest numerical values. For example,

```
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. predict ProbM ProbB ProbC ProbW ProbP
(option p assumed; predicted probabilities)
```

The variables created by `predict` are

```
. desc Prob*
```

variable name	storage type	display format	value label	variable label
ProbM	float	%9.0g		Pr(occ==1)
ProbB	float	%9.0g		Pr(occ==2)
ProbC	float	%9.0g		Pr(occ==3)
ProbW	float	%9.0g		Pr(occ==4)
ProbP	float	%9.0g		Pr(occ==5)

```
. summarize Prob*
```

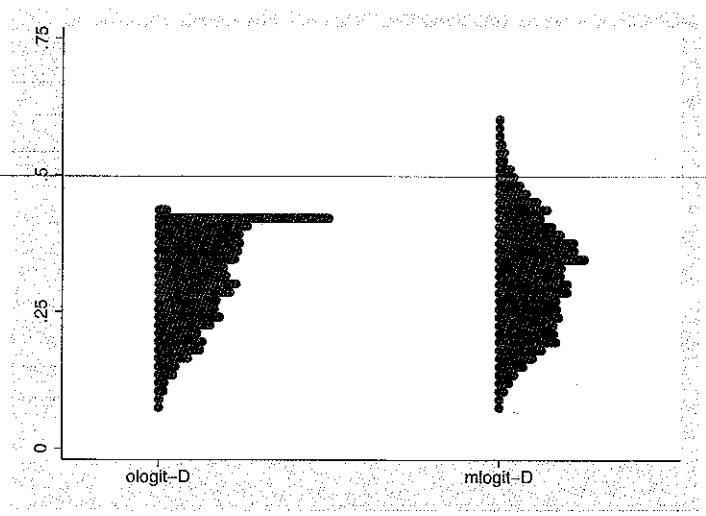
Variable	Obs	Mean	Std. Dev.	Min	Max
ProbM	337	.0919881	.059396	.0010737	.3281906
ProbB	337	.2047478	.1450568	.0012066	.6974148
ProbC	337	.2492582	.1161309	.0079713	.551609
ProbW	337	.1216617	.0452844	.0083857	.2300058
ProbP	337	.3323442	.2870992	.0001935	.9597512

Using predict to compare mlogit and ologit

An interesting way to illustrate how predictions can be plotted is to compare predictions from ordered logit and multinomial logit when the models are applied to the same data. Recall from chapter 5 that the range of the predicted probabilities for middle categories abruptly ended, whereas predictions for the end categories had a more gradual distribution. To illustrate this point, the example in chapter 5 is estimated using `ologit` and `mlogit`, with predicted probabilities computed for each case:

```
. use http://www.stata-press.com/data/lf2/ordwarm2,clear
(77 & 89 General Social Survey)
. ologit warm yr89 male white age ed prst, nolog
  (output omitted)
. predict SDologit Dologit Aologit SAologit
(option p assumed; predicted probabilities)
. label var Dologit "ologit-D"
. mlogit warm yr89 male white age ed prst, nolog
  (output omitted)
. predict SDmlogit Dmlogit Amlogit SAmlogit
(option p assumed; predicted probabilities)
. label var Dmlogit "mlogit-D"
```

We can plot the predicted probabilities of disagreeing in the two models with the command `dotplot Dologit Dmlogit, ylabel(0(.25).75)`, which leads to



Although the two sets of predictions have a correlation of .92 (computed by the command `correlate Dologit Dmlogit`), the abrupt truncation of the distribution for the ordered logit model strikes us as substantively unrealistic.

6.6.3 Predicted probabilities and discrete change with prvalue

Predicted probabilities for individuals with specified characteristics can be computed with `prvalue`. For example, we might compute the probabilities of each occupational outcome to compare nonwhites and whites who are average on education and experience:

```
. use http://www.stata-press.com/data/lf2/nomocc2, clear
(1982 General Social Survey)
. mlogit occ white ed exper, baseoutcome(5) nolog
  (output omitted)
. quietly prvalue, x(white=0) rest(mean) save
. prvalue, x(white=1) rest(mean) diff
mlogit: Change in Predictions for occ
Confidence intervals by delta method
                      Current     Saved    Change    95% CI for Change
  Pr(y=Menial|x):      0.0860    0.2168   -0.1309   [-0.3056,   0.0439]
  Pr(y=BlueCol|x):     0.1862    0.1363    0.0498   [-0.0897,   0.1893]
  Pr(y=Craft|x):       0.2790    0.4387   -0.1597   [-0.3686,   0.0491]
  Pr(y=WhiteCol|x):    0.1674    0.0877    0.0797   [-0.0477,   0.2071]
  Pr(y=Prof|x):        0.2814    0.1204    0.1611   [ 0.0277,   0.2944]
             white          ed       exper
Current=         1   13.094955   20.501484
  Saved=         0   13.094955   20.501484
   Diff=         1           0           0
```

This example also shows how to use `prvalue` to compute differences between two sets of probabilities. Our first call of `prvalue` is done `quietly`, but we `save` the results. The second call uses the `diff` option, and the output compares the results for the first and second set of values computed. By using `prvalue` with the `save` and `diff` options, we obtain confidence intervals for the discrete changes. The predicted difference between blacks and whites in the probability of having professional jobs is the only case in which the 95% confidence interval does not include zero.

6.6.4 Tables of predicted probabilities with prtab

If you want predicted probabilities for all combinations of a set of categorical independent variables, `prtab` is useful. For example, we might want to know how white and nonwhite respondents differ in their probability of having a menial job by years of education:

(Continued on next page)

```
. label def lwhite 0 NonWhite 1 White
. label val white lwhite
. prtab ed white, novarlbl outcome(1)
mlogit: Predicted probabilities of outcome 1 (Menial) for occ
```

ed	white NonWhite	White
3	0.2847	0.1216
6	0.2987	0.1384
7	0.2988	0.1417
8	0.2963	0.1431
9	0.2906	0.1417
10	0.2814	0.1366
11	0.2675	0.1265
12	0.2476	0.1104
13	0.2199	0.0883
14	0.1832	0.0632
15	0.1393	0.0401
16	0.0944	0.0228
17	0.0569	0.0120
18	0.0310	0.0060
19	0.0158	0.0029
20	0.0077	0.0014

```
        white         ed      exper
x=  .91691395  13.094955  20.501484
```

Tip: outcome() option Here we use the `outcome()` option to restrict the output to one outcome category. Without this option, `prtab` will produce a separate table for each outcome category.

The table produced by `prtab` shows the substantial differences between whites and nonwhites in the probabilities of having menial jobs and how these probabilities are affected by years of education. However, given the number of categories for `ed`, plotting these predicted probabilities with `prgen` is probably a more useful way to examine the results.

6.6.5 Graphing predicted probabilities with prgen

Predicted probabilities can be plotted using the same methods considered for the ordinal regression model. After fitting the model, we use `prgen` to compute the predicted probabilities for whites with average working experience as education increases from 6 years to 20 years:


```
. prgen ed, x(white=1) from(6) to(20) generate(wht) ncases(15)
mlogit: Predicted values as ed varies from 6 to 20.
          white         ed      exper
x=            1  13.094955  20.501484
```

Here is what the options specify:

`x(white=1)` sets `white` to 1. Because the `rest()` option is not included, all other variables are set to their means by default.

`from(6)` and `to(20)` set the minimum and maximum values over which `ed` is to vary. The default is to use the variable's minimum and maximum values.

`ncases(15)` indicates that 15 evenly spaced values of `ed` between 6 and 20 are to be generated. We chose 15 for the number of values from 6 to 20, inclusive.

`gen(wht)` specifies the root name for the new variables generated by `prgen`. For example, the variable `whtx` contains values of `ed`, the p-variables (e.g., `whtp2`) contain the predicted probabilities for each outcome, and the s-variables contain the summed probabilities:

probability of whites to be in menial job

```
. desc wht*

              storage  display     value
variable name   type   format      label      variable label

whtx            float  %9.0g                  Years of education
whtp1           float  %9.0g                  pr(Menial)=Pr(1)
whtp2           float  %9.0g                  pr(BlueCol)=Pr(2)
whtp3           float  %9.0g                  pr(Craft)=Pr(3)
whtp4           float  %9.0g                  pr(WhiteCol)=Pr(4)
whtp5           float  %9.0g                  pr(Prof)=Pr(5)
whts1           float  %9.0g                  pr(y<=1)
whts2           float  %9.0g                  pr(y<=2)
whts3           float  %9.0g                  pr(y<=3)
whts4           float  %9.0g                  pr(y<=4)
whts5           float  %9.0g                  pr(y<=5)
```

The same thing can be done to compute predicted probabilities for nonwhites:

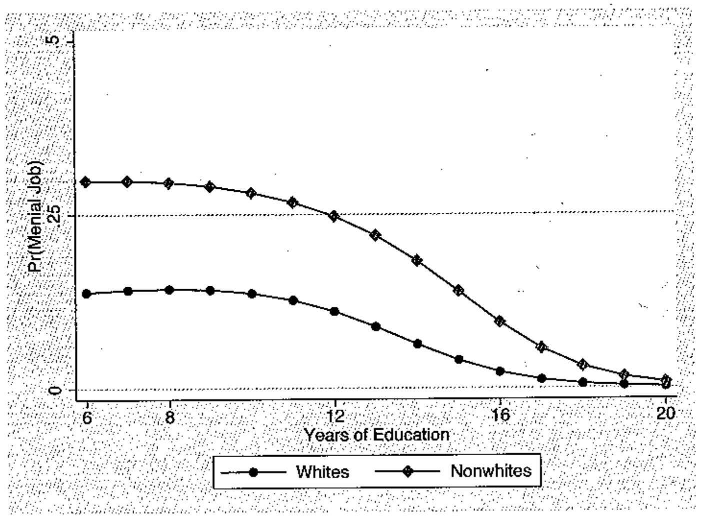
```
. prgen ed, x(white=0) from(6) to(20) generate(nwht) ncases(15)
mlogit: Predicted values as ed varies from 6 to 20.
          white         ed      exper
x=            0  13.094955  20.501484
```

Plotting probabilities for one outcome and two groups

The variables `nwhtp1` and `whtp1` contain the predicted probabilities of having menial jobs for nonwhites and whites. Plotting these provides clearer information than the results of `prtab` given above:

years of education

```
. label var whtp1 "Whites"
. label var nwhtp1 "Nonwhites"
. graph twoway connected whtp1 nwhtp1 nwhtx,
    xtitle("Years of Education")
    ytitle("Pr(Menial Job)")
    ylabel(0(.25).50) xlabel(6 8 12 16 20)
```



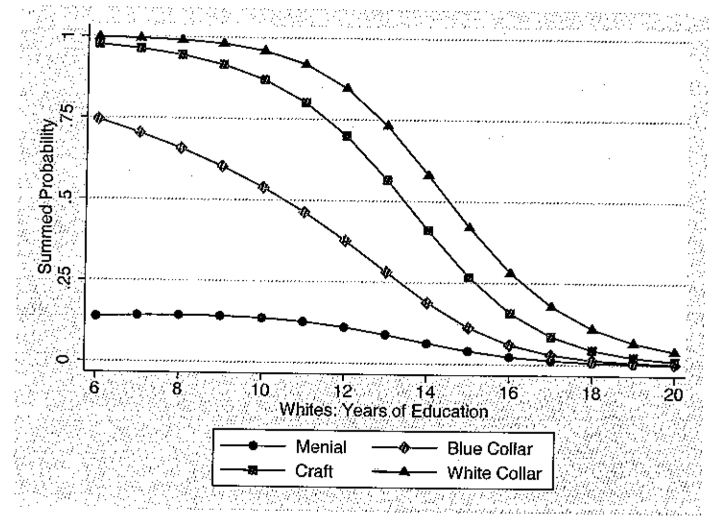
Graphing probabilities for all outcomes for one group

Even though nominal outcomes are not ordered, plotting the summed probabilities can be a useful way to show predicted probabilities for all outcome categories. To show this, we construct a graph to show how education affects the probability of each occupation for whites (a similar graph could be plotted for nonwhites). This is done using the *roots#* variables created by `prgen`, which provide the probability of being in an outcome less than or equal to some value. For example, the label for `whts3` is `pr(y<=3)`, which indicates that all nominal categories coded as 3 or less are added together. To plot these probabilities, the first thing we do is change the variable labels to the name of the highest category in the sum, which makes the graph clearer (as you will see below):

```
. label var whts1 "Menial"
. label var whts2 "Blue Collar"
. label var whts3 "Craft"
. label var whts4 "White Collar"
```


To create the summed plot, we use the following command:

```
. graph twoway connected whts1 whts2 whts3 whts4 whtx, ///
> xtitle("Whites: Years of Education")                 ///
> ytitle("Summed Probability")                         ///
> xlabel(6(2)20)                                       ///
> ylabel(0(.25)1)
```



The graph plots the four summed probabilities against `whtx`, where standard options for `graph` are used. This graph is not ideal, but before revising it, let's make sure we understand what is being plotted. The lowest line with circles, labeled "Menial" in the key, plots the probability of having a menial job for a given year of education. This is the same information as plotted in our prior graph for whites. The next line with small diamonds, labeled "Blue Collar" in the key, plots the sum of the probability of having a menial job or a blue-collar job. Thus the area between the line with circles and the line with diamonds is the probability of having a blue-collar job, and so on.

Because what we really want to illustrate are the regions between the curves, this graph is not as effective as we would like. In the `graph` command below, we use the `rarea` plot type to shade the regions between the curves. The syntax for an `rarea` plot[2] is

`graph twoway rarea` *y1var y2var xvar* [*if*] [*in*] [, *rarea_options*]

where *y1var* defines the lower boundary and *y2var* defines the upper boundary of the region for each x-value given in the variable *xvar*.

Continuing with our example, as the probabilities are bounded between zero and one, we begin by creating variables that hold these extreme values.

2. Type `help twoway rarea` for more information.

```
. gen zero = 0
. gen one  = 1
```

Now we are ready to draw the full graph.

```
. graph twoway (rarea zero whts1 whtx, bc(gs1))                    ///
>              (rarea whts1 whts2 whtx, bc(gs4))                   ///
>              (rarea whts2 whts3 whtx, bc(gs8))                   ///
>              (rarea whts3 whts4 whtx, bc(gs11))                  ///
>              (rarea whts4 one whtx, bc(gs14)),                   ///
>              ytitle("Summed Probability")                        ///
>              legend( order( 1 2 3 4 5)                           ///
>              label( 1 "Menial")                                  ///
>              label( 2 "Blue Collar") label( 3 "Craft")           ///
>              label(4 "White Collar") label(5 "Professional"))    ///
>              xtitle("Whites: Years of Education")                ///
>              xlabel(6 8 12 16 20) ylabel(0(.25)1)                ///
>              plotregion(margin(zero))
```

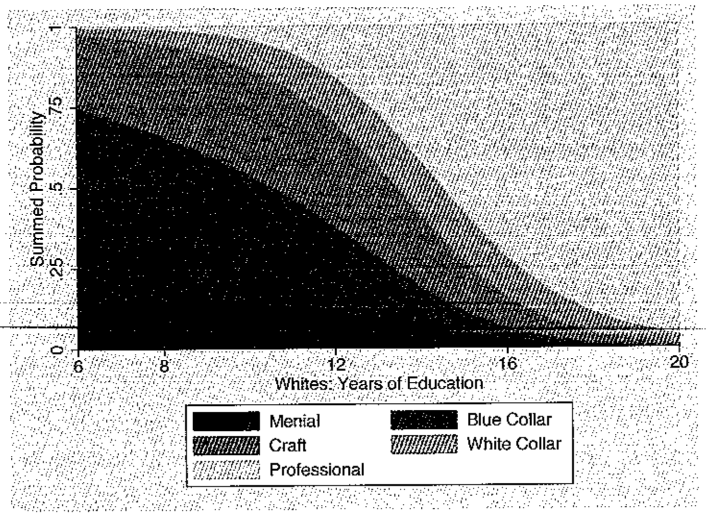


Figure 6.1: Whites: years of education.

The changes in the shaded regions in figure 6.1 clearly illustrate how the probability of selecting any one occupation changes as education increases.

6.6.6 Changes in predicted probabilities

Marginal and discrete change can be used in the same way as in models for ordinal outcomes. As before, both can be computed using prchange.

Marginal change is defined as

$$\frac{\partial \Pr(y=m\mid \mathbf{x})}{\partial x_k} = \Pr(y=m\mid \mathbf{x})\left\{\beta_{k,m|J} - \sum_{j=1}^{J}\beta_{k,j|J}\Pr(y=j\mid \mathbf{x})\right\}$$

As this equation combines all the $\beta_{k,j|J}$s, the value of the marginal change depends on the levels of all variables in the model. Further, as the value of x_k changes, the sign of the marginal can change. For example, at one point the marginal effect of education on having a craft occupation could be positive, whereas at another point the marginal effect could be negative.

Discrete change is defined as

$$\frac{\Delta \Pr(y=m\mid \mathbf{x})}{\Delta x_k} = \Pr(y=m\mid \mathbf{x}, x_k = x_E) - \Pr(y=m\mid \mathbf{x}, x_k=x_S)$$

where the magnitude of the change depends on the levels of all variables and the size of the change that is being made. The J discrete-change coefficients for a variable (one for each outcome category) can be summarized by computing the average of the *absolute values* of the changes across all the outcome categories,

$$\overline{\Delta} = \frac{1}{J}\sum_{j=1}^{J}\left|\frac{\Delta \Pr(y=j\mid \overline{\mathbf{x}})}{\Delta x_k}\right|$$

where the absolute value is taken because the sum of the changes without taking the absolute value is necessarily zero.

Computing marginal and discrete change with prchange

Discrete and marginal changes are computed with prchange (the full syntax for which is provided in chapter 3). For example,

(*Continued on next page*)

```
. mlogit occ white ed exper
  (output omitted)
. prchange
mlogit: Changes in Probabilities for occ
white
            Avg|Chg|      Menial     BlueCol       Craft    WhiteCol
    0->1  .11623582  -.13085523   .04981799  -.15973434   .07971004
               Prof
    0->1   .1610615
ed
            Avg|Chg|      Menial     BlueCol       Craft    WhiteCol
Min->Max  .39242268  -.13017954  -.70077323  -.15010394   .02425591
   -+1/2  .05855425  -.02559762  -.06831616  -.05247185   .01250795
  -+sd/2   .1640657  -.07129153  -.19310513  -.14576758   .03064777
MargEfct  .05894859  -.02579097  -.06870635  -.05287415   .01282041
               Prof
Min->Max  .95680079
   -+1/2  .13387768
  -+sd/2  .37951647
MargEfct  .13455107
exper
            Avg|Chg|      Menial     BlueCol       Craft    WhiteCol
Min->Max  .12193559  -.11536534  -.18947365   .03115708   .09478889
   -+1/2  .00233425  -.00226997  -.00356567   .00105992    .0016944
  -+sd/2  .03253578  -.03167491  -.04966453   .01479983   .02360725
MargEfct  .00233427  -.00226997  -.00356571   .00105992   .00169442
               Prof
Min->Max  .17889298
   -+1/2  .00308132
  -+sd/2  .04293236
MargEfct  .00308134
              Menial     BlueCol       Craft    WhiteCol        Prof
Pr(y|x)    .09426806   .18419114   .29411051   .16112968   .26630062
            white        ed     exper
    x=   .916914    13.095   20.5015
sd(x)=   .276423   2.94643   13.9594
```

The first thing to notice is the output labeled `Pr(y|x)`, which is the predicted probabilities at the values set by `x()` and `rest()`. Marginal change is listed in the rows `MargEfct`. For variables that are not binary, discrete change is reported over the range of the variable (reported as `Min->Max`), for changes of one unit centered on the base values (reported as `-+1/2`), and for changes of one standard deviation centered on the base values (reported as `-+sd/2`). If the `uncentered` option is used, the changes begin at the value specified by `x()` or `rest()` and increase one unit or one standard deviation from there. For binary variables, the discrete change from 0 to 1 is the only appropriate quantity and is the only quantity that is presented. Looking at the results for `white` above, we can see that for someone who is average in education and experience, the predicted probability of having a professional job is .16 higher for whites than nonwhites. The average change is listed in the column `Avg|Chg|`. For example, for `white`, $\overline{\Delta} = 0.12$, the average absolute change in the probability of various occupational categories for being white as opposed to nonwhite is .12.

Marginal change with mfx

The marginal change can also be computed using `mfx`, where the `at()` option is used to set values of the independent variables. Like `prchange`, the `mfx` command sets all values of the independent variables to their means by default. Also we must estimate the marginal effects for one outcome at a time, using the `predict(outcome(#))` option to specify the outcome for which we want marginal effects:

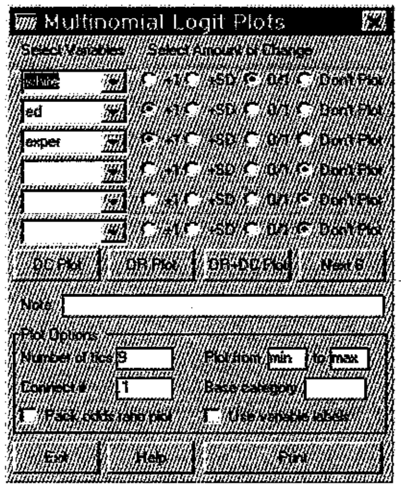
```
. mfx, predict(outcome(1))
Marginal effects after mlogit
      y  = Pr(occ==1) (predict, outcome(1))
         =  .09426806
------------------------------------------------------------------------------
variable |      dy/dx    Std. Err.     z    P>|z|  [    95% C.I.   ]      X
---------+--------------------------------------------------------------------
  white* |  -.1308552      .08914   -1.47   0.142  -.305562  .043852   .916914
      ed |   -.025791      .00688   -3.75   0.000  -.039269 -.012312    13.095
   exper |    -.00227      .00126   -1.80   0.071  -.004737  .000197   20.5015
------------------------------------------------------------------------------
(*) dy/dx is for discrete change of dummy variable from 0 to 1
```

These results are for the `Menial` category (`occ==1`). Estimates for `exper` and `ed` match the results in the `MargEfct` rows of the `prchange` output above. Meanwhile, for the binary variable `white`, the discrete change from 0 to 1 is presented, which also matches the corresponding result from `prchange`. An advantage of `mfx` is that standard errors for the effects are also provided; a disadvantage is that `mfx` can take a long time to produce results after `mlogit`, especially if the number of observations and independent variables is large.

6.6.7 Plotting discrete changes with prchange and mlogview

One difficulty with nominal outcomes is the many coefficients that need to be considered: one for each variable times the number of outcome categories minus one. To help you sort out all this information, discrete-change coefficients can be plotted using our program `mlogview`. After fitting the model with `mlogit` and computing discrete changes with `prchange`, executing `mlogview` opens the following dialog box:

(*Continued on next page*)



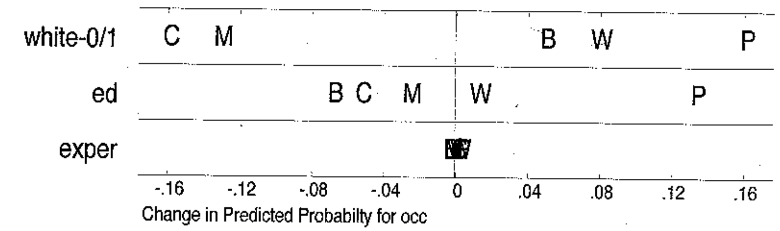
Dialog boxes are easier to use than to explain. So, as we describe various features, the best advice is to generate the dialog box shown above and experiment.

Selecting variables If you click and hold a button, you can select a variable to be plotted. The same variable can be plotted more than once, for example, showing the effects of different amounts of change.

Selecting the amount of change The radio buttons allow you to select the type of discrete-change coefficient to plot for each selected variable: +1 selects coefficients for a change of one unit; +SD selects coefficients for a change of one standard deviation; 0/1 selects changes from 0 to 1; and Don't Plot is self-explanatory.

Making a plot Even though there are more options to explain, you should try plotting your selections by clicking on DC Plot, which produces a graph. The command `mlogview` works by generating the syntax for the command `mlogplot`, which actually draws the plot. In the Results window, you will see the `mlogplot` command that was used to generate your graph (full details on `mlogplot` are given in section 6.6.9). If there is an error in the options you select, the error message will appear in the Results window.

On the assumption that everything has worked, we generate the following graph:

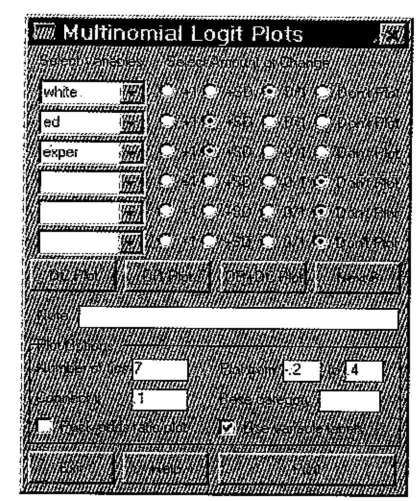


The graph immediately shows how a unit increase in each variable affects the probability of each outcome. Although it appears that the effects of being white are the largest, changes of one unit in education and (especially) experience are often too small to be as informative. It would make more sense to look at the effects of a standard deviation change in these variables. To do this, we return to the dialog box and click on the radio button +SD. Before we see what this does, let's consider several other options that can be used.

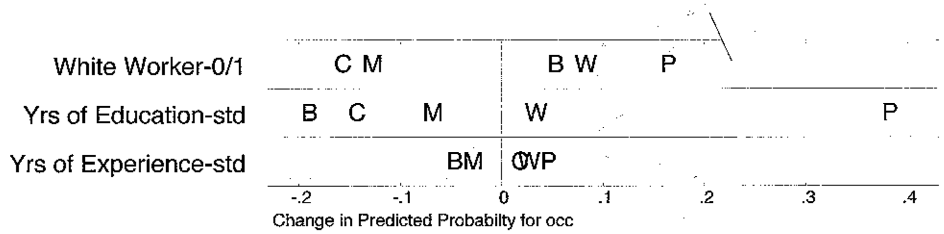
Adding labels The box Note allows you to enter text that will be placed at the top of the graph. Clicking the box for Use variable labels replaces the names of the variables on the left axis with the variable labels associated with each variable. When you do this, you may find that the labels are too long. If so, you can use the `label variable` command to change them.

Tick marks The values for the tick marks are determined by specifying the minimum and maximum values to plot and the number of tick marks. For example, we could specify a plot from −.2 to .4 with seven tick marks. This will lead to labels every .1 units.

Using some of the features discussed above, our dialog box would look like this:



Clicking on DC Plot produces the following graph:



You can see that the effects of education are largest and that those of experience are smallest. Or, each coefficient can be interpreted individually, such as the following:

> The effects of a standard deviation change in education are largest, with an increase of more than .35 in the probability of having a professional occupation.
>
> The effects of race are also substantial, with average blacks being less likely to enter blue-collar, white-collar, or professional jobs than average whites.
>
> Expected changes due to a standard deviation change in experience are much smaller and show that experience increases the probabilities of more highly skilled occupations.

In using these graphs, remember that different values for discrete change are obtained at different levels of the variables, which are specified with the `x()` and `rest()` options for `prchange`.

Value labels with mlogview The value labels for the different categories of the dependent variables must begin with different letters because the plots generated with `mlogview` use the first letter of the value label.

6.6.8 Odds ratios using listcoef and mlogview

Discrete change does little to illuminate the dynamics among the outcomes. For example, a decrease in education increases the probability of both blue-collar and craft jobs, but how does it affect the odds of a person choosing a craft job relative to a blue-collar job? To deal with these issues, odds ratios (also referred to as factor change coefficients) can be used. Holding other variables constant, the factor change in the odds of outcome m versus outcome n as x_k increases by δ equals

$$\frac{\Omega_{m|n}(\mathbf{x}, x_k+\delta)}{\Omega_{m|n}(\mathbf{x}, x_k)} = e^{\beta_{k,m|n}\delta}$$

If the amount of change is $\delta = 1$, the odds ratio can be interpreted as follows:

> For a unit change in x_k, the odds of m versus n are expected to change by a factor of $\exp(\beta_{k,m|n})$, holding all other variables constant.

If the amount of change is $\delta = s_{x_k}$, then the odds ratio can be interpreted as follows:

> For a standard deviation change in x_k, the odds of m versus n are expected to change by a factor of $\exp(\beta_{k,m|n} \times s_k)$, holding all other variables constant.

Listing odds ratios with listcoef

The difficulty in interpreting odds ratios for the MNLM is that, to understand the effect of a variable, you need to examine the coefficients for comparisons among all pairs of outcomes. The standard output from `mlogit` includes only $J-1$ comparisons with the base category. Although you could estimate coefficients for all possible comparisons by rerunning `mlogit` with different base categories (e.g., `mlogit occ white ed exper, baseoutcome(3)`), using `listcoef` is much simpler. For example, to examine the effects of race, type

```
. listcoef white, help
mlogit (N=337): Factor Change in the Odds of occ
Variable: white (sd=.27642268)

Odds comparing      |
Alternative 1       |
to Alternative 2    |        b         z     P>|z|      e^b    e^bStdX
--------------------+------------------------------------------------
Menial   -BlueCol   |  -1.23650   -1.707    0.088   0.2904    0.7105
Menial   -Craft     |  -0.47234   -0.782    0.434   0.6235    0.8776
Menial   -WhiteCol  |  -1.57139   -1.741    0.082   0.2078    0.6477
Menial   -Prof      |  -1.77431   -2.350    0.019   0.1696    0.6123
BlueCol  -Menial    |   1.23650    1.707    0.088   3.4436    1.4075
BlueCol  -Craft     |   0.76416    1.208    0.227   2.1472    1.2352
BlueCol  -WhiteCol  |  -0.33488   -0.359    0.720   0.7154    0.9116
BlueCol  -Prof      |  -0.53780   -0.673    0.501   0.5840    0.8619
Craft    -Menial    |   0.47234    0.782    0.434   1.6037    1.1395
Craft    -BlueCol   |  -0.76416   -1.208    0.227   0.4657    0.8096
Craft    -WhiteCol  |  -1.09904   -1.343    0.179   0.3332    0.7380
Craft    -Prof      |  -1.30196   -2.011    0.044   0.2720    0.6978
WhiteCol-Menial     |   1.57139    1.741    0.082   4.8133    1.5440
WhiteCol-BlueCol    |   0.33488    0.359    0.720   1.3978    1.0970
WhiteCol-Craft      |   1.09904    1.343    0.179   3.0013    1.3550
WhiteCol-Prof       |  -0.20292   -0.233    0.815   0.8163    0.9455
Prof     -Menial    |   1.77431    2.350    0.019   5.8962    1.6331
Prof     -BlueCol   |   0.53780    0.673    0.501   1.7122    1.1603
Prof     -Craft     |   1.30196    2.011    0.044   3.6765    1.4332
Prof     -WhiteCol  |   0.20292    0.233    0.815   1.2250    1.0577
---------------------------------------------------------------------

       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
```

The odds ratios of interest are in the column labeled `e^b`. For example, the odds ratio for the effect of race on having a professional versus a menial job is 5.90, which can be interpreted as follows:

> The odds of having a professional occupation relative to a menial occupation are 5.90 times greater for whites than for blacks, holding education and experience constant.

Remember: the `gt`, `lt`, and `pvalue` options control which comparisons are printed by `listcoef`. See pages 233–234 for more details.

Plotting odds ratios

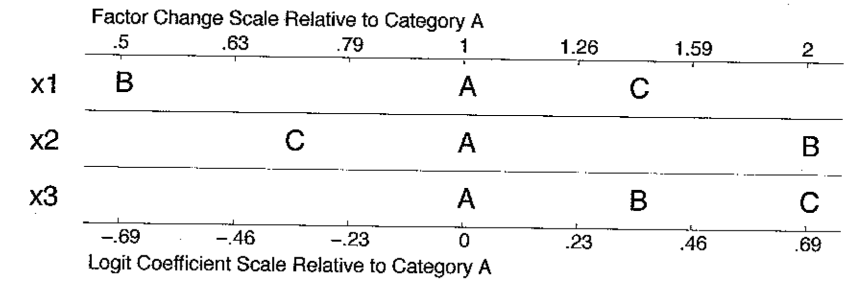
However, examining all the coefficients for even a single variable with only five dependent categories is complicated. An *odds-ratio plot* makes it easy to quickly see patterns in results for even a complex MNLM (see Long 1997, chapter 6 for full details). To explain how to interpret an odds ratio plot, we begin with some hypothetical output from a MNLM with three outcomes and three independent variables:

		Logit coefficient for:		
Comparison		x_1	x_2	x_3
$B \mid A$	$\beta_{B\mid A}$	−0.693	0.693	0.347
	$\exp(\beta_{B\mid A})$	0.500	2.000	1.414
	p	0.04	0.01	0.42
$C \mid A$	$\beta_{C\mid A}$	0.347	−0.347	0.693
	$\exp(\beta_{C\mid A})$	1.414	0.707	2.000
	p	0.21	0.04	0.37
$C \mid B$	$\beta_{C\mid B}$	1.040	−1.040	0.346
	$\exp(\beta_{C\mid B})$	2.828	0.354	1.414
	p	0.02	0.03	0.21

These coefficients were constructed to have some fixed relationships among categories and variables:

- The effects of x_1 and x_2 on $B \mid A$ (which you can read as B versus A) are equal but of opposite size. The effect of x_3 is half as large.
- The effects of x_1 and x_2 on $C \mid A$ are half as large (and in opposite directions) as the effects on $B \mid A$, whereas the effect of x_3 is in the same direction but twice as large.

In the odds-ratio plot, the independent variables are each represented on a separate row, and the horizontal axis indicates the relative magnitude of the β coefficients associated with each outcome. Here is the plot, where the letters correspond to the outcome categories:



The plot reveals much information, which we now summarize.

Sign of coefficients

If a letter is to the right of another letter, increases in the independent variable make the outcome to the right more likely. Thus relative to outcome A, an increase in x_1 makes it more likely that we will observe outcome C and less likely that we will observe outcome B. This corresponds to the positive sign of the $\beta_{1,C\mid A}$ coefficient and the negative sign of the $\beta_{1,B\mid A}$ coefficient. The signs of these coefficients are reversed for x_2, and accordingly, the odds-ratio plot for x_2 is a mirror image of that for x_1.

Magnitude of effects

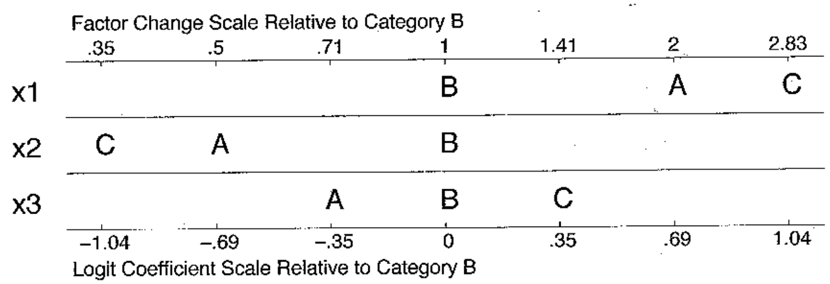
The distance between a pair of letters indicates the magnitude of the effect. For both x_1 and x_2, the distance between A and B is twice the distance between A and C, which reflects that $\beta_{B\mid A}$ is twice as large as $\beta_{C\mid A}$ for both variables. For x_3, the distance between A and B is half the distance between A and C, reflecting that $\beta_{3,C\mid A}$ is twice as large as $\beta_{3,B\mid A}$.

The additive relationship

The additive relationships among coefficients shown in (6.1) are also fully reflected in this graph. For any of the independent variables, $\beta_{C\mid A} = \beta_{B\mid A} + \beta_{C\mid B}$. Accordingly, the distance from A to C is the sum of the distances from A to B and B to C.

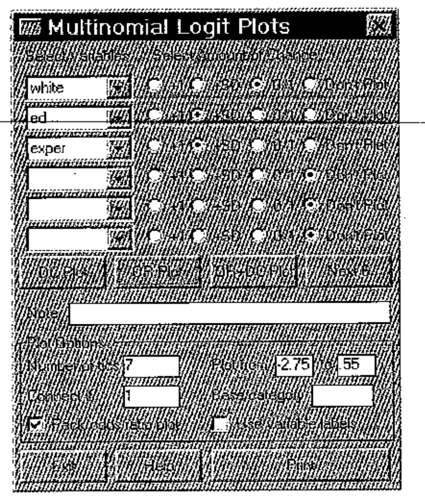
The base category

The additive scale on the bottom axis measures the value of the $\beta_{k,m|n}$s. The multiplicative scale on the top axis measures the $\exp\left(\beta_{k,m|n}\right)$s. The As are stacked on top of one another because the plot uses A as its base category for graphing the coefficients. The choice of base category is arbitrary. We could have used alternative B instead. If we had, the rows of the graph would be shifted to the left or right so that the Bs lined up. Doing this leads to the following graph:

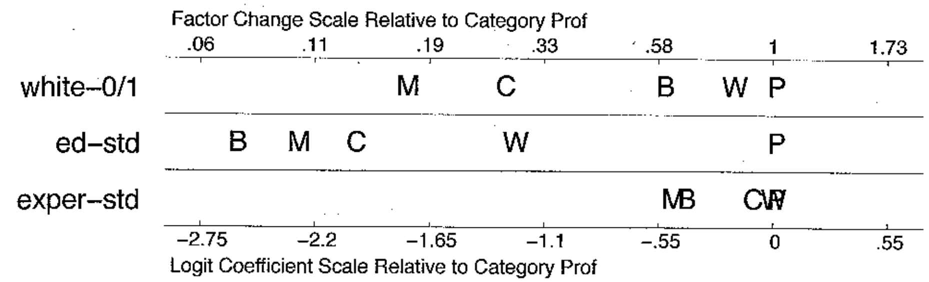


Creating odds-ratio plots

These graphs can be created using `mlogview` after running `mlogit`. Using our example and after changing a few options, we obtain this dialog box:



Clicking on OR Plot gives

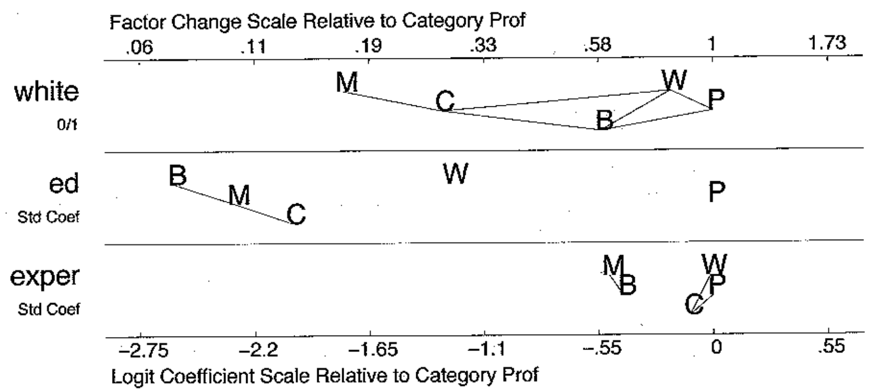


Several things are immediately apparent. The effect of experience is the smallest, although increases in experience make it more likely that one will be in a craft, white-collar, or professional occupation relative to a menial or blue-collar one. We also see that education has the largest effect; as expected, increases in education increase the odds of having a professional job relative to any other type.

Adding significance levels

The current graph does not reflect statistical significance. This is added by drawing a line between categories for which there is *not* a significant coefficient. The *lack* of statistical significance is shown by a connecting line, suggesting that those two outcomes are "tied together". You can add the significance level to the plot with the Connect if box on the dialog box. For example, if we enter .1 in this box and uncheck the "pack odds ratio plot" box, we obtain

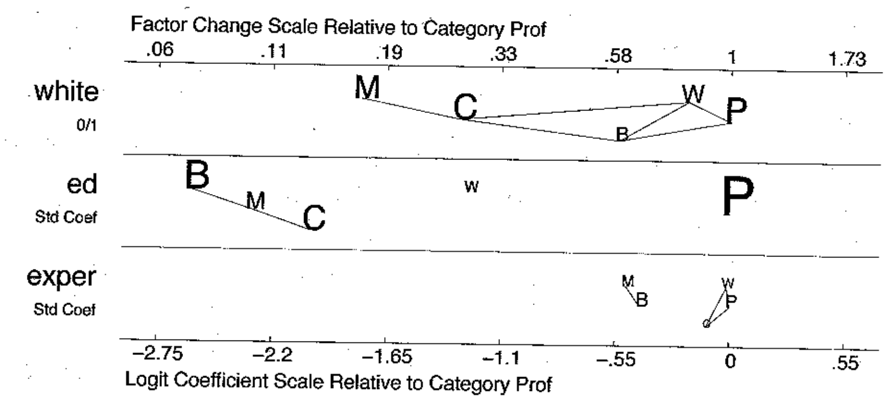
(*Continued on next page*)



To make the connecting lines clear, vertical spacing is added to the graph. *This vertical spacing has no meaning and is used only to make the lines clearer.* The graph shows that race orders occupations from menial to craft to blue collar to white collar to professional, but the connecting lines show that none of the adjacent categories are significantly differentiated by race. Being white increases the odds of being a craft worker relative to having a menial job, but the effect is not significant. However, being white significantly increases the odds of being a blue-collar worker, a white-collar worker, or a professional, relative to having a menial job. The effects of `ed` and `exper` can be interpreted similarly.

Adding discrete change

In chapter 4, we emphasized that *whereas the factor change in the odds is constant across the levels of all variables, the discrete change gets larger or smaller at different values of the variables.* For example, if the odds increase by a factor of 10 but the current odds are 1 in 10,000, the substantive impact is small. But if the current odds were 1 in 5, the impact is large. Information on the discrete change in probability can be incorporated in the odds-ratio graph by making the size of the letter proportional to the discrete change in the odds (specifically, the area of the letter is proportional to the size of the discrete change). This can easily be added to our graph. First, after estimating the MNLM, run `prchange` at the levels of the variables that you want. Then enter `mlogview` to open the dialog box. Set any of the options, and then click the OR+DC Plot button:

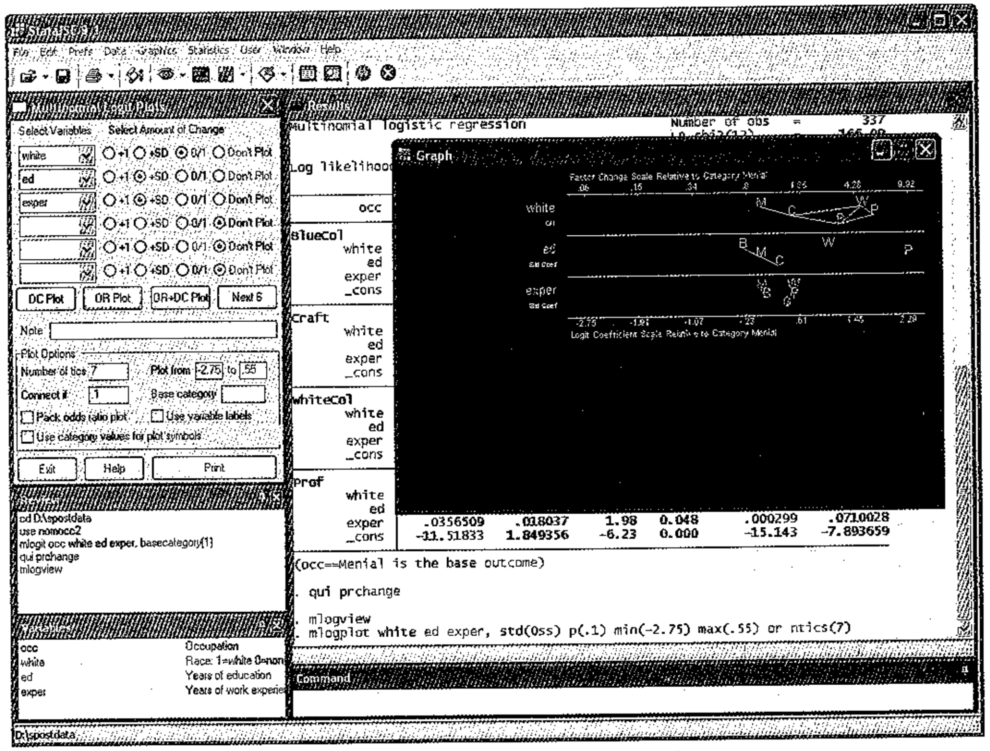


With a little practice, you can quickly create and interpret these graphs.

6.6.9 Using mlogplot*

The dialog box `mlogview` does not actually draw the plots but only sends the options you select to `mlogplot`, which creates the graph. Once you click a plot button in `mlogview`, the necessary `mlogplot` command, including options, appears in the Results window. This is done because `mlogview` invokes a dialog box and so cannot be used effectively in a do-file. But once you create a plot using the dialog box you can copy the generated `mlogplot` command from the Results window and paste it into a do-file. This should be clear by looking at the following screenshot:

(*Continued on next page*)



The dialog box with selected options appears in the upper left of the screen. After we clicked on the OR Plot button, the graph in the upper right appeared along with the following command in the Results window:

```
. mlogplot white ed exper, std(0ss) p(.1) min(-2.75) max(.55) or ntics(7)
```

If you enter this command from the Command window or run it from a do-file, the same graph will be generated. The full syntax for mlogplot is described in appendix A.

6.6.10 Plotting estimates from matrices with mlogplot*

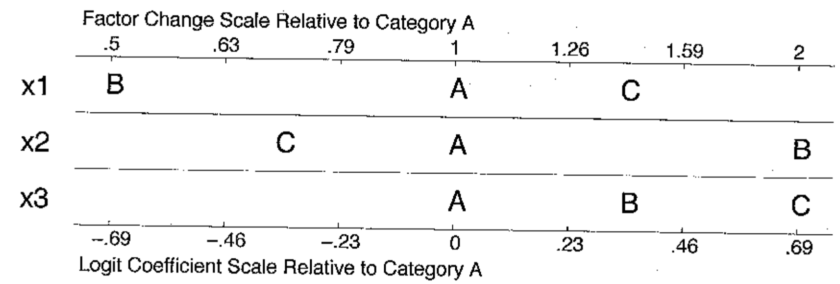
You can also use mlogplot to construct odds-ratio plots (but not discrete-change plots) using coefficients that are to be contained in matrices. For example, you can plot coefficients from published papers or generate examples like those we used above. To do this, you must construct matrices containing the information to be plotted and add the option matrix to the command. The easiest way to see how this is done is with an example, followed by details on each matrix. The commands

```
. matrix mnlbeta = (-.693, .693, .347  .347, -.347, .693 )
. matrix mnlsd   = (1, 2, 4)
. global mnlname = "x1 x2 x3"
```



```
. global mnlcatnm = "B C A"
. global mnldepnm "depvar"
. mlogplot, matrix std(uuu) vars(x1 x2 x3) packed
```

create the following plot:



Options for using matrices with mlogplot

matrix indicates that the coefficients to be plotted are contained in matrices.

vars(*varlist*) contains the names of the variables to be plotted. This list must contain names from mnlname, which will be described next, but does not need to be in the same order as in mnlname. The list can contain the same name more than once and can select a subset of the names from mnlname.

Global macros and matrices used by mlogplot

mnlname is a string containing the names of the variables corresponding to the columns of the matrix mnlbeta. For example, global mnlname = "x1 x2 x3".

mnlbeta is a matrix with the βs, where element (i,j) is the coefficient $\beta_{j,i|b}$. That is, rows i are for different contrasts; columns j are for variables. For example, matrix mnlbeta = (-.693, .693, .347 \ .347, -.347, .693). As constant terms are *not* plotted, they are not included in mnlbeta.

mnlsd is a vector with the standard deviations for the variables listed in mnlname. For example, matrix mnlsd = (1, 2, 4). If you do not want to view standardized coefficients, this matrix can be made all 1s.

mnlcatnm is a string with labels for the outcome categories with each label separated by a space. For example, global mnlcatnm = "B C A". The first label corresponds to the first row of mnlbeta, the second to the second, and so on. *The label for the base category is last.*

Example

Suppose that you want to compare the logit coefficients estimated from two groups, such as whites and nonwhites from the example used in this chapter. We begin by estimating the logit coefficients for whites:

```
. use http://www.stata-press.com/data/lf2/nomocc2, clear
(1982 General Social Survey)
. mlogit occ ed exper if white==1, base(5) nolog
Multinomial logistic regression                   Number of obs   =        309
                                                  LR chi2(8)      =     154.60
                                                  Prob > chi2     =     0.0000
Log likelihood = -388.21313                       Pseudo R2       =     0.1660
```

occ	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Menial						
ed	-.8307514	.1297238	-6.40	0.000	-1.085005	-.5764973
exper	-.0338038	.0192045	-1.76	0.078	-.071444	.0038364
_cons	10.34842	1.779603	5.82	0.000	6.860465	13.83638
BlueCol						
ed	-.9225522	.1085452	-8.50	0.000	-1.135297	-.7098075
exper	-.031449	.0150766	-2.09	0.037	-.0609987	-.0018994
_cons	12.27337	1.507683	8.14	0.000	9.318368	15.22838
Craft						
ed	-.6876114	.0952882	-7.22	0.000	-.8743729	-.50085
exper	-.0002589	.0131021	-0.02	0.984	-.0259385	.0254207
_cons	9.017976	1.36333	6.61	0.000	6.345897	11.69005
WhiteCol						
ed	-.4196403	.0956209	-4.39	0.000	-.6070539	-.2322268
exper	.0008478	.0147558	0.06	0.954	-.0280731	.0297687
_cons	4.972973	1.421146	3.50	0.000	2.187578	7.758368

(occ==Prof is the base outcome)

Next we compute coefficients for nonwhites:

```
. mlogit occ ed exper if white==0, base(5) nolog
Multinomial logistic regression                   Number of obs   =         28
                                                  LR chi2(8)      =      17.79
                                                  Prob > chi2     =     0.0228
Log likelihood = -32.779416                       Pseudo R2       =     0.2135
```

occ	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Menial						
ed	-.7012628	.3331146	-2.11	0.035	-1.354155	-.0483701
exper	-.1108415	.0741488	-1.49	0.135	-.2561705	.0344876
_cons	12.32779	6.053743	2.04	0.042	.4626714	24.19291
BlueCol						
ed	-.560695	.3283292	-1.71	0.088	-1.204208	.0828185
exper	-.0261099	.0682348	-0.38	0.702	-.1598477	.1076279
_cons	8.063397	6.008358	1.34	0.180	-3.712768	19.83956
Craft						
ed	-.882502	.3359805	-2.63	0.009	-1.541012	-.2239924
exper	-.1597929	.0744172	-2.15	0.032	-.305648	-.0139378
_cons	16.21925	6.059753	2.68	0.007	4.342356	28.09615
WhiteCol						
ed	-.5311514	.369815	-1.44	0.151	-1.255976	.1936728
exper	-.0520881	.0838967	-0.62	0.535	-.2165227	.1123464
_cons	7.821371	6.805367	1.15	0.250	-5.516904	21.15965

(occ==Prof is the base outcome)

The two sets of coefficients for ed are placed in mnlbeta:

```
. matrix mnlbeta = (-.8307514, -.9225522, -.6876114, -.4196403 \
      -.7012628, -.560695 , -.882502 , -.5311514)
```

Rows of the matrix correspond to the variables (i.e., ed for whites and ed for nonwhites) since this was the easiest way to enter the coefficients. For mlogplot, the columns must correspond to variables, so we transpose the matrix:

```
. matrix mnlbeta = mnlbeta'
```

We assign names to the columns using mnlname and to the rows using mnlcatnm (where the last element is the name of the reference outcome):

```
. global mnlname = "White NonWhite"
. global mnlcatnm = "Menial BlueCol Craft WhiteCol Prof"
```

We named the coefficients for ed for whites, White, and the coefficients for ed for nonwhites, NonWhite, as this will make the plot clearer. Next we compute the standard deviation of ed:

```
. summarize ed
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
          ed |       337    13.09496    2.946427          3         20
```

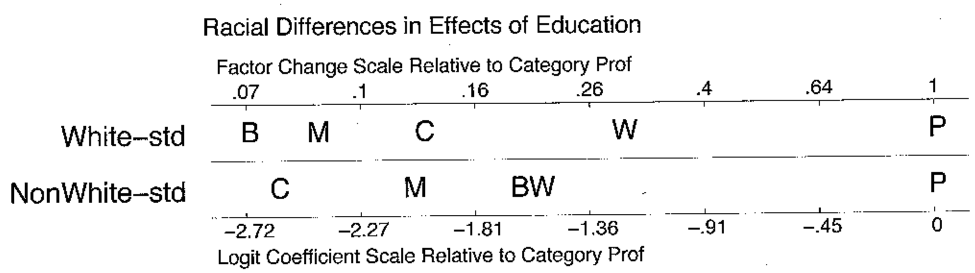
and enter the information into `mnlsd`:

```
. matrix mnlsd = (2.946427,2.946427)
```

The same value is entered twice because we want to use the overall standard deviation in education for both groups. To create the plot, we use the command

```
. mlogplot, vars(White NonWhite) packed
      or matrix std(ss)
      note("Racial Differences in Effects of Education")
```

which leads to



Given the limitations of our dataset (e.g., there were only 28 cases in the logit for nonwhites) and our simple model, these results do not represent serious research on racial differences in occupational outcomes, but they show the flexibility of the `mlogplot` command.

6.7 Multinomial probit model with IIA

The multinomial probit regression command `mprobit` is the normal error counterpart to the multinomial logit model fitted by `mlogit` in the same way that `probit` is the normal counterpart to `logit`. However, `mprobit` uses a normalization that can obscure this fact. To understand this point, we need to consider how logit and probit models can be motivated as discrete-choice models in which a person maximizes her utility.

Let u_{im} be the utility that person i receives from alternative m. The utility is assumed to be determined by a linear combination of observed characteristics $\mathbf{x}_i$ and random error ε_{im}:

$$u_{im} = \mathbf{x}_i \boldsymbol{\beta}_m + \varepsilon_{im}$$

Since the utility associated with each alternative m is partly determined by chance through ε, the model is also called a random utility model (RUM). A person chooses alternative j if the utility associated with that alternative is larger than that for any other alternative. Accordingly, the probability of alternative m being chosen is

$$\Pr(y_i = m) = \Pr(u_{im} > u_{ij} \text{ for all } j \neq m)$$

The choice that a person makes under these assumptions will not change if the utility associated with each alternative changes by some fixed amount, say, δ. That is, if $u_{im} > u_{ij}$, then $u_{im} + \delta > u_{ij} + \delta$. Thus the choice is based on the difference in the utilities between alternatives. We can incorporate this idea into the model by taking the difference in the utilities for two alternatives. To illustrate this, assume that there are three alternatives. We can consider the utility of each alternative relative to some base alternative. It does not matter which alternative is chosen as the base, so we assume that each utility is compared with alternative 1. Accordingly, we have

$$\begin{aligned} u_{i1} - u_{i1} &= 0 \\ u_{i2} - u_{i1} &= \mathbf{x}_i (\boldsymbol{\beta}_2 - \boldsymbol{\beta}_1) + (\varepsilon_{i2} - \varepsilon_{i1}) \\ u_{i3} - u_{i1} &= \mathbf{x}_i (\boldsymbol{\beta}_3 - \boldsymbol{\beta}_1) + (\varepsilon_{i3} - \varepsilon_{i1}) \end{aligned}$$

If we define $u^*_{im} \equiv u_{im} - u_{i1}$, $\varepsilon^*_{im} \equiv \varepsilon_{im} - \varepsilon_{i1}$ and $\boldsymbol{\beta}_{m|1} \equiv \boldsymbol{\beta}_m - \boldsymbol{\beta}_1$, the model can be written as:

$$\begin{aligned} u^*_{i2} &= \mathbf{x}_i \boldsymbol{\beta}_{2|1} + \varepsilon^*_{i2} \\ u^*_{i3} &= \mathbf{x}_i \boldsymbol{\beta}_{3|1} + \varepsilon^*_{i3} \end{aligned}$$

The specific form of the model depends on the distribution of the error terms. Assuming that the εs have an extreme value distribution with mean 0 and variance $\pi^2/6$ leads to the MNLM that we discussed with respect to `mlogit`. Assuming that the εs have a normal distribution leads to a probit-type model. To understand the model fitted by `mprobit` and how it relates to the usual binary probit model, we need to pay careful attention to the assumed variance of the errors. The binary probit model fitted by `probit` makes the usual assumption that $\text{Var}(\varepsilon_j) = 1/2$, so $\text{Var}(\varepsilon^*_j) = \text{Var}(\varepsilon_j) + \text{Var}(\varepsilon_1) = 1$. Since we assume that the errors are uncorrelated, $\text{Cov}(\varepsilon_j, \varepsilon_1) = 0$. Using our earlier example for labor force participation, we can fit the binary probit model:

(Continued on next page)