Title

stata.com

marginsplot — Graph results from margins (profile plots, etc.)

Description

`marginsplot` graphs the results of the immediately preceding `margins` command; see [R] **margins**. Common names for some of the graphs that `marginsplot` can produce are profile plots and interaction plots.

Menu

Statistics > Postestimation

Syntax

`marginsplot` [, *options*]

options	Description
Main	
`xdimension(`*dimlist* [, *dimopts*]`)`	use *dimlist* to define x axis
`plotdimension(`*dimlist* [, *dimopts*]`)`	create plots for groups in *dimlist*
`bydimension(`*dimlist* [, *dimopts*]`)`	create subgraphs for groups in *dimlist*
`graphdimension(`*dimlist* [, *dimopts*]`)`	create graphs for groups in *dimlist*
`horizontal`	swap x and y axes
`noci`	do not plot confidence intervals
`name(`*name* | *stub* [, `replace`]`)`	name of graph, or stub if multiple graphs
Labels	
`allxlabels`	place ticks and labels on the x axis for each value
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo variable name and equal signs in all labels
`nosimplelabels`	include variable name and equal signs in all labels
`separator(`*string*`)`	separator for labels when multiple variables are specified in a dimension
`noseparator`	do not use a separator

Plot	
`plotopts(`*plot_options*`)`	affect rendition of all margin plots
`plot#opts(`*plot_options*`)`	affect rendition of #th margin plot
`recast(`*plottype*`)`	plot margins using *plottype*
CI plot	
`ciopts(`*rcap_options*`)`	affect rendition of all confidence interval plots
`ci#opts(`*rcap_options*`)`	affect rendition of #th confidence interval plot
`recastci(`*plottype*`)`	plot confidence intervals using *plottype*
`mcompare(`*method*`)`	adjust for multiple comparisons
`level(#)`	set confidence level
Pairwise	
`unique`	plot only unique pairwise comparisons
`csort`	sort comparison categories first
Add plots	
`addplot(`*plot*`)`	add other plots to the graph
Y axis, X axis, Titles, Legend, Overall, By	
twoway_options	any options documented in [G-3] ***twoway_options***
`byopts(`*byopts*`)`	how subgraphs are combined, labeled, etc.

where *dimlist* may be any of the dimensions across which margins were computed in the immediately preceding `margins` command; see [R] **margins**. That is to say, *dimlist* may be any variable used in the `margins` command, including variables specified in the `at()`, `over()`, and `within()` options. More advanced specifications of *dimlist* are covered in *Addendum: Advanced uses of dimlist*.

dimopts	Description
`labels(`*lablist*`)`	list of quoted strings to label each level of the dimension
`elabels(`*elablist*`)`	list of enumerated labels
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo variable name and equal signs in all labels
`nosimplelabels`	include variable name and equal signs in all labels
`separator(`*string*`)`	separator for labels when multiple variables are specified in the dimension
`noseparator`	do not use a separator

where *lablist* is defined as

"*label*" ["*label*" [...]]

elablist is defined as

"*label*" [*#* "*label*" [...]]

and the #s are the indices of the levels of the dimension—1 is the first level, 2 is the second level, and so on.

plot_options	Description
marker_options	change look of markers (color, size, etc.)
marker_label_options	add marker labels; change look or position
cline_options	change look of the line

method	Description
noadjust	do not adjust for multiple comparisons
bonferroni [adjustall]	Bonferroni's method; adjust across all terms
sidak [adjustall]	Šidák's method; adjust across all terms
scheffe	Scheffé's method

Options

Main

xdimension(), plotdimension(), bydimension(), and graphdimension() specify the variables from the preceding margins command whose group levels will be used for the graph's x axis, plots, by() subgraphs, and graphs.

marginsplot chooses default dimensions based on the margins command. In most cases, the first variable appearing in an at() option and evaluated over more than one value is used for the x axis. If no at() variable meets this condition, the first variable in the *marginlist* is usually used for the x axis and the remaining variables determine the plotted lines or markers. Pairwise comparisons and graphs of marginal effects (derivatives) have different defaults. In all cases, you may override the defaults and explicitly control which variables are used on each dimension of the graph by using these dimension options.

Each of these options supports suboptions that control the labeling of the dimension—axis labels for xdimension(), plot labels for plotdimension(), subgraph titles for bydimension(), and graph titles for graphdimension() titles.

For examples using the dimension options, see *Controlling the graph's dimensions*.

xdimension(*dimlist* [, *dimopts*]) specifies the variables for the x axis in *dimlist* and controls the content of those labels with *dimopts*.

plotdimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the variables whose group levels determine the plots and optionally specifies in *dimopts* the content of the plots' labels.

bydimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the variables whose group levels determine the by() subgraphs and optionally specifies in *dimopts* the content of the subgraphs' titles. For an example using by(), see *Three-way interactions*.

graphdimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the variables whose group levels determine the graphs and optionally specifies in *dimopts* the content of the graphs' titles.

horizontal reverses the default x and y axes. By default, the y axis represents the estimates of the margins and the x axis represents one or more factors or continuous covariates. Specifying horizontal swaps the axes so that the x axis represents the estimates of the margins. This option can be useful if the labels on the factor or continuous covariates are long.

The horizontal option is discussed in *Horizontal is sometimes better*.

noci removes plots of the pointwise confidence intervals. The default is to plot the confidence intervals.

`name(`*name* | *stub* [`, replace`]`)` specifies the name of the graph or graphs. If the `graphdimension()` option is specified, or if the default action is to produce multiple graphs, then the argument of `name()` is taken to be *stub* and graphs named *stub*`1`, *stub*`2`, ... are created.

The `replace` suboption causes existing graphs with the specified name or names to be replaced.

If `name()` is not specified, default names are used and the graphs may be replaced by subsequent `marginsplot` or other graphing commands.

Labels

With the exception of `allxlabels`, all of these options may be specified either directly as options or as *dimopts* within options `xdimension()`, `plotdimension()`, `bydimension()`, and `graphdimension()`. When specified in one of the dimension options, only the labels for that dimension are affected. When specified outside the dimension options, all labels on all dimensions are affected. Specifications within the dimension options take precedence.

`allxlabels` specifies that tick marks and labels be placed on the x axis for each value of the x-dimension variables. By default, if there are more than 25 ticks, default graph axis labeling rules are applied. Labeling may also be specified using the standard `graph twoway` x-axis label rules and options—`xlabel()`; see [G-3] ***axis_label_options***.

`nolabels` specifies that value labels not be used to construct graph labels and titles for the group levels in the dimension. By default, if a variable in a dimension has value labels, those labels are used to construct labels and titles for axis ticks, plots, subgraphs, and graphs.

Graphs of contrasts and pairwise comparisons are an exception to this rule and are always labeled with values rather than value labels.

`allsimplelabels` and `nosimplelabels` control whether graphs' labels and titles include just the values of the variables or include variable names and equal signs. The default is to use just the value label for variables that have value labels and to use variable names and equal signs for variables that do not have value labels. An example of the former is "Female" and the latter is "country=2".

Sometimes value labels are universally descriptive, and sometimes they have meaning only when considered in relation to their variable. For example, "Male" and "Female" are typically universal, regardless of the variable from which they are taken. "High" and "Low" may not have meaning unless you know they are in relation to a specific measure, say, blood-pressure level. The `allsimplelabels` and `nosimplelabels` options let you override the default labeling.

`allsimplelabels` specifies that all titles and labels use just the value or value label of the variable.

`nosimplelabels` specifies that all titles and labels include *varname*`=` before the value or value label of the variable.

`separator(`*string*`)` and `noseparator` control the separator between label sections when more than one variable is used to specify a dimension. The default separator is a comma followed by a space, but no separator may be requested with `noseparator` or the default may be changed to any string with `separator()`.

For example, if `plotdimension(a b)` is specified, the plot labels in our graph legend might be "a=1, b=1", "a=1, b=2", ... Specifying `separator(:)` would create labels "a=1:b=1", "a=1:b=2", ...

Plot

`plotopts(`*plot_options*`)` affects the rendition of all margin plots. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

These settings may be overridden for specific plots by using the `plot#opts()` option.

`plot#opts(`*plot_options*`)` affects the rendition of the #th margin plot. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

`recast(`*plottype*`)` specifies that margins be plotted using *plottype*. *plottype* may be `scatter`, `line`, `connected`, `bar`, `area`, `spike`, `dropline`, or `dot`; see [G-2] **graph twoway**. When `recast()` is specified, the plot-rendition options appropriate to the specified *plottype* may be used in lieu of *plot_options*. For details on those options, follow the appropriate link from [G-2] **graph twoway**.

For an example using `recast()`, see *Continuous covariates*.

You may specify `recast()` within a `plotopts()` or `plot#opts()` option. It is better, however, to specify it as documented here, outside those options. When specified outside those options, you have greater access to the plot-specific rendition options of your specified *plottype*.

CI plot

`ciopts(`*rcap_options*`)` affects the rendition of all confidence interval plots; see [G-3] ***rcap_options***.

These settings may be overridden for specific confidence interval plots with the `ci#opts()` option.

`ci#opts(`*rcap_options*`)` affects the rendition of the #th confidence interval; see [G-3] ***rcap_options***.

`recastci(`*plottype*`)` specifies that confidence intervals be plotted using *plottype*. *plottype* may be `rarea`, `rbar`, `rspike`, `rcap`, `rcapsym`, `rline`, `rconnected`, or `rscatter`; see [G-2] **graph twoway**. When `recastci()` is specified, the plot-rendition options appropriate to the specified *plottype* may be used in lieu of *rcap_options*. For details on those options, follow the appropriate link from [G-2] **graph twoway**.

For an example using `recastci()`, see *Continuous covariates*.

You may specify `recastci()` within a `ciopts()` or `ci#opts()` option. It is better, however, to specify it as documented here, outside those options. When specified outside those options, you have greater access to the plot-specific rendition options of your specified *plottype*.

`mcompare(`*method*`)` specifies the method for confidence intervals that account for multiple comparisons within a factor-variable term. The default is determined by the `margins` results stored in `r()`. If `marginsplot` is working from `margins` results stored in `e()`, the default is `mcompare(noadjust)`.

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is determined by the `margins` results stored in `r()`. If `marginsplot` is working from `margins` results stored in `e()`, the default is `level(95)` or as set by `set level`; see **[U] 20.8 Specifying the width of confidence intervals**.

Pairwise

These options have an effect only when the `pwcompare` option was specified on the preceding `margins` command.

`unique` specifies that only unique pairwise comparisons be plotted. The default is to plot all pairwise comparisons, including those that are mirror images of each other—"male" versus "female" and "female" versus "male". `margins` reports only the unique pairwise comparisons. `unique`

also changes the default `xdimension()` for graphs of pairwise comparisons from the reference categories (`_pw0`) to the comparisons of each pairwise category (`_pw`).

Unique comparisons are often preferred with horizontal graphs that put all pairwise comparisons on the x axis, whereas including the full matrix of comparisons is preferred for charts showing the reference groups on an axis and the comparison groups as plots; see *Pairwise comparisons* and *Horizontal is sometimes better*.

`csort` specifies that comparison categories are sorted first, and then reference categories are sorted within comparison category. The default is to sort reference categories first, and then sort comparison categories within reference categories. This option has an observable effect only when `_pw` is also specified in one of the dimension options. It then determines the order of the labeling in the dimension where `_pw` is specified.

Add plots

`addplot(`*plot*`)` provides a way to add other plots to the generated graph; see [G-3] ***addplot_option***.

For an example using `addplot()`, see *Adding scatterplots of the data*.

If multiple graphs are drawn by a single `marginsplot` command or if *plot* specifies plots with multiple y variables, for example, `scatter y1 y2 x`, then the graph's legend will not clearly identify all the plots and will require customization using the `legend()` option; see [G-3] ***legend_options***.

Y axis, X axis, Titles, Legend, Overall, By

twoway_options are any of the options documented in [G-3] ***twoway_options***. These include options for titling the graph (see [G-3] ***title_options***); for saving the graph to disk (see [G-3] ***saving_option***); for controlling the labeling and look of the axes (see [G-3] ***axis_options***); for controlling the look, contents, position, and organization of the legend (see [G-3] ***legend_options***); for adding lines (see [G-3] ***added_line_options***) and text (see [G-3] ***added_text_options***); and for controlling other aspects of the graph's appearance (see [G-3] ***twoway_options***).

The `label()` suboption of the `legend()` option has no effect on `marginsplot`. Use the `order()` suboption instead.

`byopts(`*byopts*`)` affects the appearance of the combined graph when `bydimension()` is specified or when the default graph has subgraphs, including the overall graph title, the position of the legend, and the organization of subgraphs. See [G-3] ***by_option***.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Introduction

`marginsplot` is a post-`margins` command. It graphs the results of the `margins` command, whether those results are marginal means, predictive margins, marginal effects, contrasts, pairwise comparisons, or other statistics; see [R] **margins**.

By default, the margins are plotted on the y axis, and all continuous and factor covariates specified in the `margins` command will usually be placed on the x axis or used to identify plots. Exceptions are discussed in the following sections and in *Addendum: Advanced uses of dimlist* below.

`marginsplot` produces classic plots, such as profile plots and interaction plots. Beyond that, anything that `margins` can compute, `marginsplot` can graph.

We will be using some relatively complicated `margins` commands with little explanation of the syntax. We will also avoid lengthy interpretations of the results of margins. See [R] **margins** for the complete syntax of `margins` and discussions of its results.

All graphs in this entry were drawn using the `s2gcolor` scheme; see [G-4] **scheme s2**.

Mitchell (2012) shows in many examples how to use `marginsplot` to understand a fitted model.

Dataset

For continuity, we will use one dataset for most examples—the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981). NHANES II is part of a study to assess the health and nutritional status of adults and children in the United States. It is designed to be a nationally representative sample of the U.S. population. This particular sample is from 1976 to 1980.

The survey nature of the dataset—weights, strata, and sampling units—will be ignored in our analyses. We are discussing graphing, not survey statistics. If you would like to see the results with the appropriate adjustments for the survey design, just add `svy:` before each estimation command, and if you wish, add `vce(unconditional)` as an option to each `margins` command. See [R] **margins**, particularly the discussion and examples under *Obtaining margins with survey data and representative*

samples, for reasons why you probably would want to add `vce(unconditional)` when analyzing survey data. For the most part, adjusting for survey design produces moderately larger confidence intervals and relatively small changes in point estimates.

Profile plots

What does my estimation say about how my response varies as one (or more) of my covariates changes? That is the question that is answered by profile plots. Profile plots are also referred to as plots of estimated (or expected, or least-squares) means, though that is unnecessarily restrictive when considering models of binary, count, and ordered outcomes. In the latter cases, we might prefer to say they plot conditional expectations of responses, where a response might be a probability.

What we do with the other covariates depends on the questions we wish to answer. Sometimes we wish to hold other covariates at fixed values, and sometimes we wish to average the response over their values. `margins` can do either, so you can graph either.

We can fit a fully factorial two-way ANOVA of systolic blood pressure on age group and sex using the NHANES II data.

```
. use http://www.stata-press.com/data/r15/nhanes2
. anova bpsystol agegrp##sex
```

```
                  Number of obs =     10,351    R-squared     =  0.2497
                  Root MSE      =    20.2209    Adj R-squared =  0.2489
          Source | Partial SS         df         MS        F     Prob>F
-----------------+----------------------------------------------------
           Model |  1407229.3         11   127929.93    312.88  0.0000
                 |
          agegrp |  1243037.8          5   248607.56    608.02  0.0000
             sex |  27728.379          1   27728.379     67.81  0.0000
  agegrp#sex     |  88675.043          5   17735.009     43.37  0.0000
                 |
        Residual |  4227440.7     10,339   408.88294
-----------------+----------------------------------------------------
           Total |    5634670     10,350   544.41256
```

If you are more comfortable with regression than ANOVA, then type

```
. regress bpsystol agegrp##sex
```

The `anova` and `regress` commands fit identical models. The output from `anova` displays all the terms in the model and thus tends to be more conducive to exploration with `margins` and `marginsplot`.

We estimate the predictive margins of systolic blood pressure for each age group using `margins`.

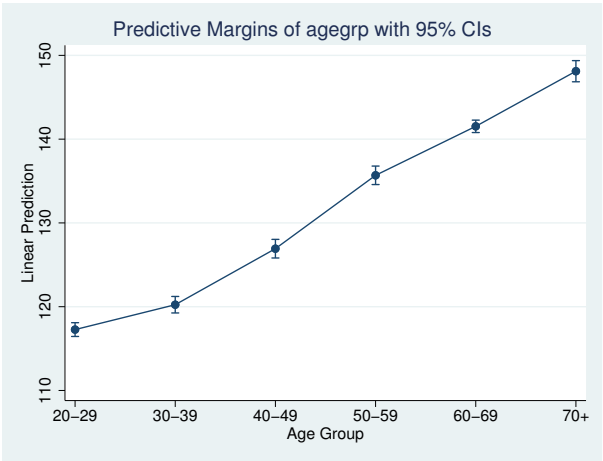
```
. margins agegrp
Predictive margins                              Number of obs     =     10,351
Expression   : Linear prediction, predict()
```

	Margin	Delta-method Std. Err.	t	P>|t|	[95% Conf.	Interval]
agegrp						
20-29	117.2684	.419845	279.31	0.000	116.4454	118.0914
30-39	120.2383	.5020813	239.48	0.000	119.2541	121.2225
40-49	126.9255	.56699	223.86	0.000	125.8141	128.0369
50-59	135.682	.5628593	241.06	0.000	134.5787	136.7853
60-69	141.5285	.3781197	374.30	0.000	140.7873	142.2696
70+	148.1096	.6445073	229.80	0.000	146.8463	149.373

The six predictive margins are just the averages of the predictions over the estimation sample, holding `agegrp` to each of its six levels. If this were a designed experiment rather than survey data, we might wish to assume the cells are balanced—that they have the same number of observations—and thus estimate what are often called expected means or least-squares means. To do that, we would simply add the `asbalanced` option to the `margins` command. The NHANES II data are decidedly unbalanced over `sex#agegrp` cells. So much so that it is unreasonable to assume the cells are balanced.

We graph the results:

```
. marginsplot
  Variables that uniquely identify margins: agegrp
```



Profile plots are often drawn without confidence intervals (CIs). The CIs may be removed by adding the `noci` option. We prefer to see the CIs.

Disciplines vary widely in their use of the term profile plot. Some disciplines consider any connected plot of a response over values of other variables to be a profile plot. By that definition, most graphs in this entry are profile plots.

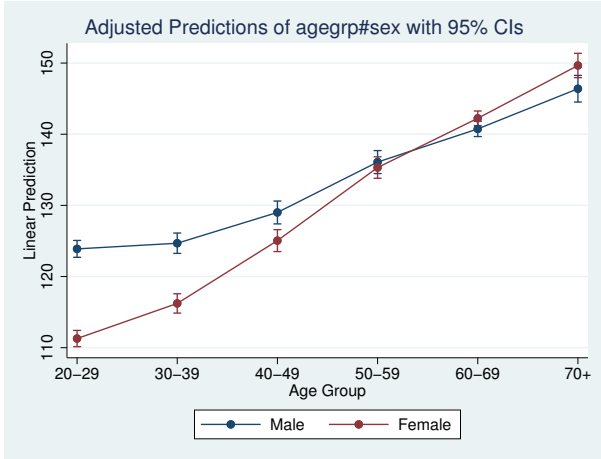
Interaction plots

Interaction plots are often used to explore the form of an interaction. The interaction term in our ANOVA results is highly significant. Are the interaction effects also large enough to matter? What form do they take? We can answer these questions by fixing `agegrp` and `sex` to each possible combination of the two covariates and estimating the margins for those cells.

```
. margins agegrp#sex
```

Then we can graph the results:

```
. marginsplot
  Variables that uniquely identify margins: agegrp sex
```



It is clear that the effect of age differs by sex—there is an interaction. If there were no interaction, then the two lines would be parallel.

While males start out with higher systolic blood pressure, females catch up to the males as age increases and may even surpass males in the upper age groups. We say "may" because we cannot tell if the differences are statistically significant. The CIs overlap for the top three age groups. It is tempting to conclude from this overlap that the differences are not statistically significant. Do not fall into this trap. Likewise, do not fall into the trap that the first three age groups are different because their CIs do not overlap. The CIs are for the point estimates, not the differences. There is a covariance between the differences that we must consider if we are to make statements about those differences.

Contrasts of margins—effects (discrete marginal effects)

To assess the differences, all we need to do is ask `margins` to contrast the sets of effects that we just estimated; see [R] **margins, contrast**. With only two groups in sex, it does not matter much which contrast operator we choose. We will use the reference contrast. It will compare the difference between males and females, with males (the first category) as the reference category.

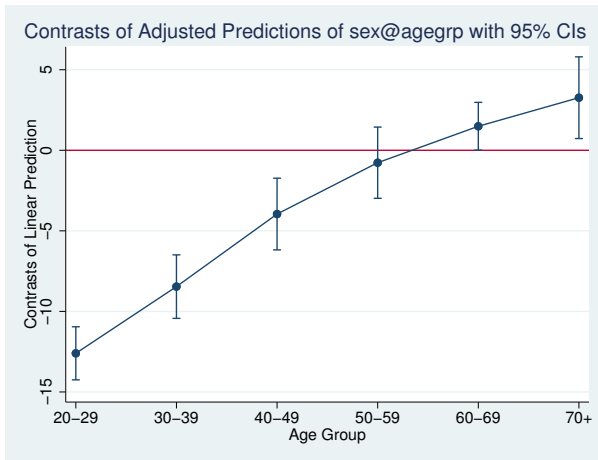
```
. margins r.sex@agegrp
Contrasts of adjusted predictions
Expression   : Linear prediction, predict()
```

	df	F	P>F
sex@agegrp			
(Female vs Male) 20-29	1	224.92	0.0000
(Female vs Male) 30-39	1	70.82	0.0000
(Female vs Male) 40-49	1	12.15	0.0005
(Female vs Male) 50-59	1	0.47	0.4949
(Female vs Male) 60-69	1	3.88	0.0488
(Female vs Male) 70+	1	6.37	0.0116
Joint	6	53.10	0.0000
Denominator	10339		

	Contrast	Delta-method Std. Err.	[95% Conf.	Interval]
sex@agegrp				
(Female vs Male) 20-29	-12.60132	.8402299	-14.24833	-10.9543
(Female vs Male) 30-39	-8.461161	1.005448	-10.43203	-6.490288
(Female vs Male) 40-49	-3.956451	1.134878	-6.181031	-1.731871
(Female vs Male) 50-59	-.7699782	1.128119	-2.981309	1.441353
(Female vs Male) 60-69	1.491684	.756906	.0080022	2.975367
(Female vs Male) 70+	3.264762	1.293325	.729594	5.79993

Because we are looking for effects that are different from 0, we will add a reference line at 0 to our graph.

```
. marginsplot, yline(0)
  Variables that uniquely identify margins: agegrp
```



We can now say that females' systolic blood pressure is substantially and significantly lower than males' in the first three age groups but is significantly higher in the last two age groups. Despite the overlapping CIs for the last two age groups in the interaction graph, the effect of sex is significant in these age groups.

The terminology for what we just estimated and graphed varies widely across disciplines. Those versed in design of experiments refer to these values as contrasts or effects. Economists and some other social scientists call them marginal or partial effects. The latter groups might be more comfortable if we avoided the whole concept of contrasts and instead estimated the effects by typing

```
. margins agegrp, dydx(sex)
```

This will produce estimates that are identical to those shown above, and we can graph them by typing `marginsplot`.

The advantage of using the contrast notation and thinking in contrasts is most evident when we take marginal effects with respect to a categorical covariate with more than two levels. Marginal effects for each level of the covariate will be taken with respect to a specified base level. Contrasts are much more flexible. Using the `r.` operator, we can reproduce the marginal-effects results by taking derivatives with respect to a reference level (as we saw above.) We can also estimate the marginal effect of first moving from level 1 to level 2, then from level 2 to level 3, then from level 3 to level 4, ... using the `ar.` or "reverse adjacent" operator. Adjacent effects (marginal effects) can be valuable when evaluating an ordinal covariate, such as `agegrp` in our current model. For a discussion of contrasts, see [R] **contrast** and [R] **margins, contrast**.

Three-way interactions

`marginsplot` can handle any number of covariates in your `margins` command. Consider the three-way ANOVA model that results from adding an indicator for whether an individual has been diagnosed with diabetes. We will fully interact the new covariate with the others in the model.

```
. anova bpsystol agegrp##sex##diabetes
```

```
                    Number of obs =     10,349    R-squared     =  0.2572
                    Root MSE      =     20.131    Adj R-squared =  0.2556
```

Source	Partial SS	df	MS	F	Prob>F
Model	1448983.2	23	62999.268	155.45	0.0000
agegrp	107963.58	5	21592.716	53.28	0.0000
sex	1232.7927	1	1232.7927	3.04	0.0812
agegrp#sex	11679.592	5	2335.9185	5.76	0.0000
diabetes	7324.9892	1	7324.9892	18.07	0.0000
agegrp#diabetes	5484.5462	5	1096.9092	2.71	0.0189
sex#diabetes	102.98824	1	102.98824	0.25	0.6142
agegrp#sex#diabetes	4863.1497	5	972.62994	2.40	0.0349
Residual	4184296.9	10,325	405.25878		
Total	5633280	10,348	544.38346		

The three-way interaction is significant, as is the main effect of `diabetes` and its interaction with `agegrp`.

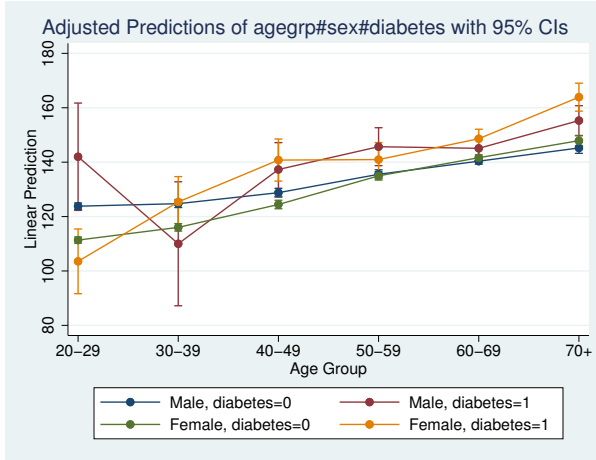
Again, if you are more comfortable with regression than ANOVA, you may type

```
. regress bpsystol agegrp##sex##diabetes
```

The `margins` and `marginsplot` results will be the same.

We estimate the expected cell means for each combination of `agegrp`, `sex`, and `diabetes`, and then graph the results by typing

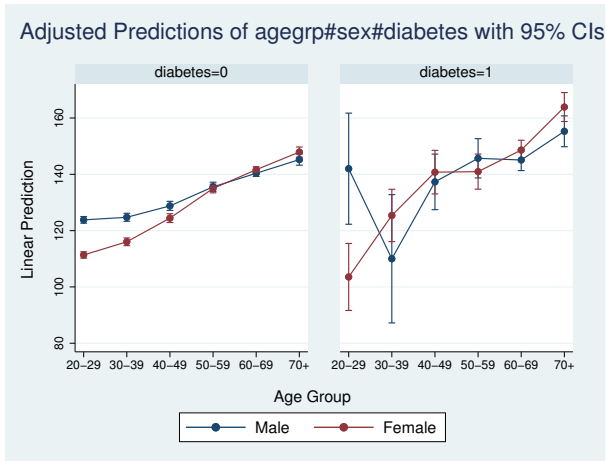
```
. margins agegrp#sex#diabetes
(output omitted)
. marginsplot
Variables that uniquely identify margins: agegrp sex diabetes
```



The graph is busy and difficult to interpret.

We can make it better by putting those with diabetes on one subgraph and those without on another:

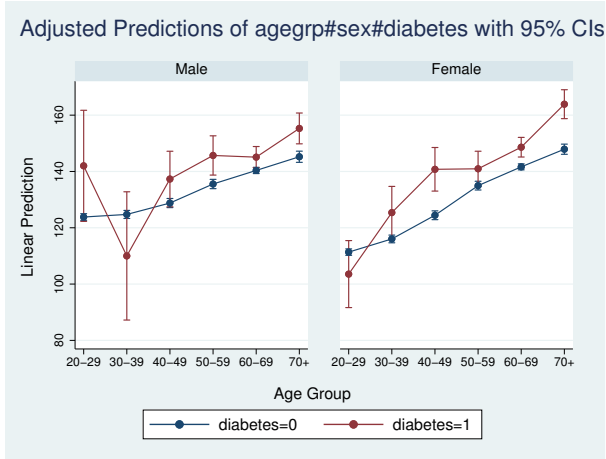
```
. marginsplot, by(diabetes)
Variables that uniquely identify margins: agegrp sex diabetes
```



We notice much larger CIs for diabetics. That is not surprising because our sample contains only 499 diabetics compared with 9,850 nondiabetics.

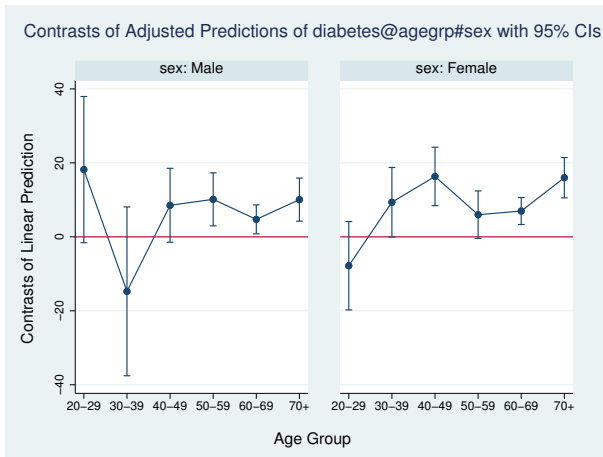
A more interesting way to arrange the plots is by grouping the subgraphs on `sex`:

```
. marginsplot, by(sex)
  Variables that uniquely identify margins: agegrp sex diabetes
```



Aside from increased systolic blood pressure in the upper-age groups, which we saw earlier, it appears that those with diabetes are at greater risk of higher systolic blood pressure for many upper-age groups. We can check that by having `margins` estimate the differences between diabetics and nondiabetics, and graphing the results.

```
. margins r.diabetes@agegrp#sex
  (output omitted)
. marginsplot, by(sex) yline(0)
  Variables that uniquely identify margins: agegrp sex
```



With CIs above 0 for six of eight age groups over 40, this graph provides evidence that diabetes is related to higher blood pressure in those over 40.

Continuous covariates

margins and marginsplot are just as useful with continuous covariates as they are with factor variables. As a variation on our ANOVA/regression models, let's move to a logistic regression, using as our dependent variable an indicator for whether a person has high blood pressure. We introduce a continuous covariate—body mass index (BMI), a measure of weight relative to height. High BMI is often associated with high blood pressure. We will allow the effect of BMI to vary across sexes, age groups, and sex/age combinations by fully interacting the covariates.

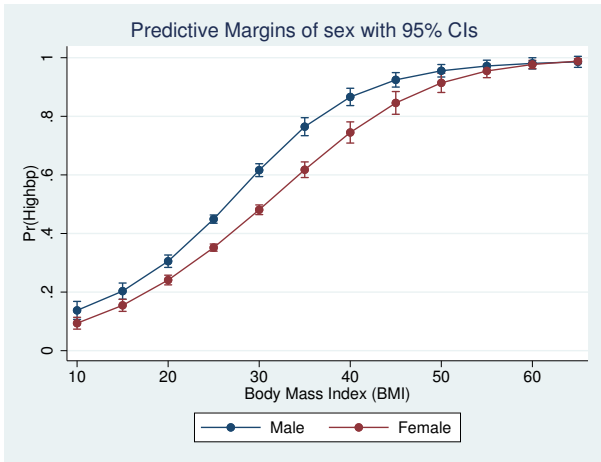
```
. logistic highbp sex##agegrp##c.bmi
```

If we wished, we could perform all the analyses above on this model. Instead of estimating margins, contrasts, and marginal effects on the level of systolic blood pressure, we would be estimating margins, contrasts, and marginal effects on the probability of having high blood pressure. You can see those results by repeating any of the prior commands that involve sex and agegrp. In this section, we will focus on the continuous covariate bmi.

With continuous covariates, rather than specify them in the *marginlist* of margins, we specify the specific values at which we want the covariate evaluated in an at() option. at() options are very flexible, and there are many ways to specify values; see *Syntax of at()* in [R] **margins**.

BMI in our sample ranges from 12.4 to 61.1. Let's estimate the predictive margins for males and females at levels of BMI from 10 through 65 at intervals of 5 and graph the results:

```
. margins sex, at(bmi=(10(5)65))
(output omitted)
. marginsplot, xlabel(10(10)60)
Variables that uniquely identify margins: bmi sex
```

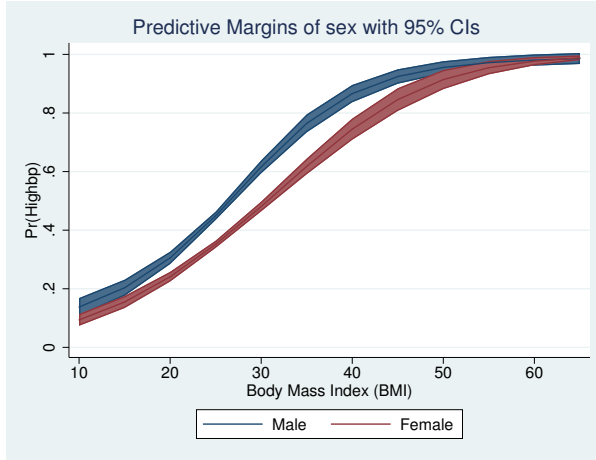


We added the xlabel(10(10)60) option to improve the labeling of the x axis. You may add any *twoway_options* (see [G-3] ***twoway_options***) to the marginsplot command.

For a given BMI, males are generally more susceptible to high blood pressure, though the effect is attenuated by the logistic response when the probabilities approach 0 or 1.

Because bmi is continuous, we might prefer to see the response graphed using a line. We might also prefer that the CIs be plotted as areas. We change the plottype of the response by using the recast() option and the plottype of the CI by using the recastci() option:

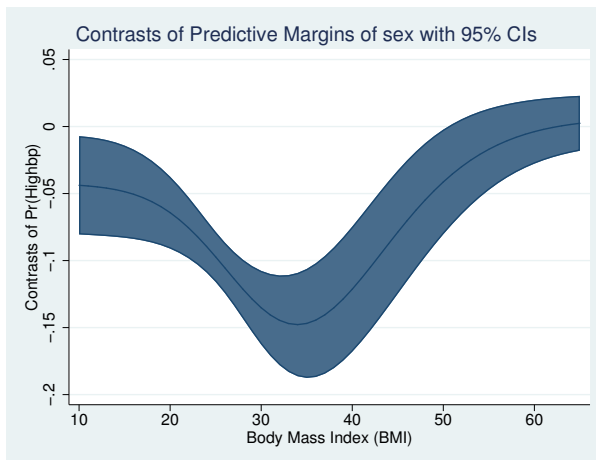
```
. marginsplot, xlabel(10(10)60) recast(line) recastci(rarea)
  Variables that uniquely identify margins: bmi sex
```



The CIs are a little dark for our tastes. You can dim them a bit by reducing the intensity of their color. Adding `ciopts(color(*.8))` to our `marginsplot` command will do that. Any plot option accepted by `twoway rarea` (see [G-2] **graph twoway rarea**) may be specified in a `ciopts()` option.

Given their confidence regions, the male and female profiles appear to be statistically different over most of the range of BMI. As with the profiles of categorical covariates, we can check that assertion by contrasting the two profiles on `sex` and graphing the results. Let's improve the smoothness of the response by specifying intervals of 1 instead of 5.

```
. margins r.sex, at(bmi=(10(1)65))
  (output omitted)
. marginsplot, xlabel(10(10)60) recast(line) recastci(rarea)
  Variables that uniquely identify margins: bmi
```



We see that the difference between the sexes is largest at a BMI of about 35 and that the sexes respond more similarly with very high and very low BMI. This shape is largely determined by the

response of the logistic function, which is attenuated near probabilities 0 and 1, combined with the fact that the lowest measured BMIs are associated with low probabilities of high blood pressure and the highest measured BMIs are associated with high probabilities of high blood pressure.

As when we contrasted profiles of categorical variables, different disciplines will think of this graph differently. Those familiar with designed experiments will be comfortable with the terms used above—this is a contrast of profiles, or a profile of effects, or a profile of a contrast. Many social scientists will prefer to think of this as a graph of marginal or partial effects. For them, this is a plot of the discrete marginal effect of being female for various levels of BMI. They can obtain an identical graph, with labeling more appropriate for the marginal effect's interpretation, by typing

```
. margins, at(bmi=(10(1)65)) dydx(sex)
. marginsplot, xlabel(10(10)60) recast(line) recastci(rarea)
```

We can also plot profiles of the response of BMI by levels of another continuous covariate (rather than by the categorical variable `sex`). To do so, we will need another continuous variable in our model. We have been using age groups as a covariate to emphasize the treatment of categorical variables and to allow the effect of age to be flexible. Our dataset also has age recorded in integer years. We replace `agegrp` with continuous `age` in our logistic regression.

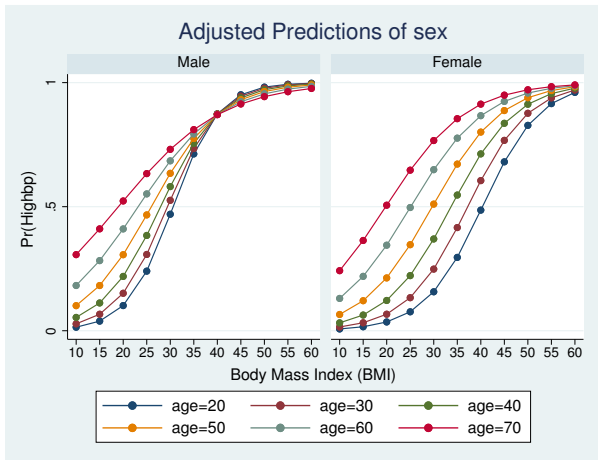
```
. logistic highbp sex##c.age##c.bmi
```

We can now obtain profiles of BMI for different ages by specifying ranges for both `bmi` and `age` in a single `at()` option on the `margins` command:

```
. margins sex, at(bmi=(10(5)60) age=(20(10)70))
```

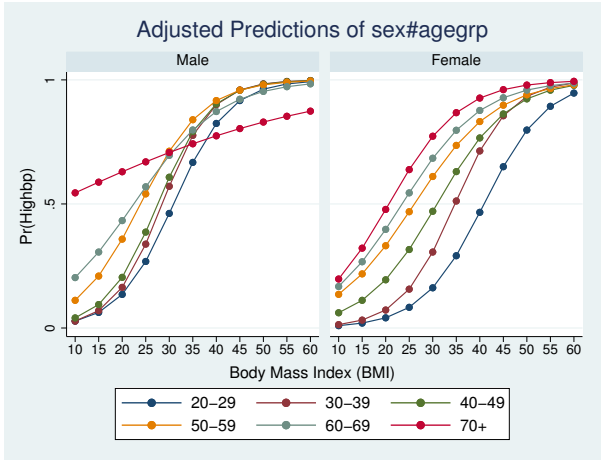
With six ages specified, we have many profiles, so we will dispense with the CIs by adding the `noci` option and also tidy up the graph by asking for three columns in the legend:

```
. marginsplot, noci by(sex) legend(cols(3))
  Variables that uniquely identify margins: bmi age sex
```



Our model seems to indicate that males have a sharper reaction to body mass indices than do females. Likewise, younger subjects display a sharper response, while older subjects have a more gradual response with earlier onset. That interpretation might be a result of our parametric treatment of `age`. As it turns out, the interpretation holds if we allow age to take more flexible forms or return to our use of age groups, which allows each of seven age groups to have unique BMI profiles. Here are the commands to perform that analysis:

```
. logistic highbp sex##agegrp##c.bmi
  (output omitted)
. margins sex#agegrp, at(bmi=(10(5)60))
  (output omitted)
. marginsplot, noci by(sex) legend(cols(3))
  Variables that uniquely identify margins: bmi sex agegrp
```



Plots at every value of a continuous covariate

In some cases, the specific values of a continuous covariate are important, and we want to plot the response at those specific values. Return to our logistic example with age treated as a continuous covariate.

```
. logistic highbp sex##c.age##c.bmi
```

We can use a programming trick to extract all the values of age and then supply them in an `at()` option, just as we would any list of values.

```
. levelsof age
. margins sex, at(age=(`r(levels)'))
```

See [P] **levelsof** for a discussion of the `levelsof` command. `levelsof` returns in `r(levels)` the sorted list of unique values of the specified *varlist*, in our case, `age`.

We can then plot the results using `marginsplot`.

This is not a very interesting trick when using our `age` variable, which is recorded as integers from 20 to 74, but the approach will work with almost any continuous variable. In our model, `bmi` might seem more interesting, but there are 9,941 unique values of `bmi` in our dataset. A graph cannot resolve so many different values. For that reason, we usually recommend against plotting at every value of a covariate. Instead, graph at reasonable values over the range of the covariate by using the `at()` option, as we did earlier. This trick is best reserved for variables with a few, or at most a few dozen, unique values.

Contrasts of at() groups—discrete effects

We have previously contrasted across the values of factor variables in our model. Put another way, we have estimated the discrete marginal effects of factor variables. We can do the same for the levels of variables in `at()` specifications and across separate `at()` specifications.

Returning to one of our logistic models and its margins, we earlier estimated the predictive margins of BMI at 5-unit intervals for both sexes. These are the commands we typed:

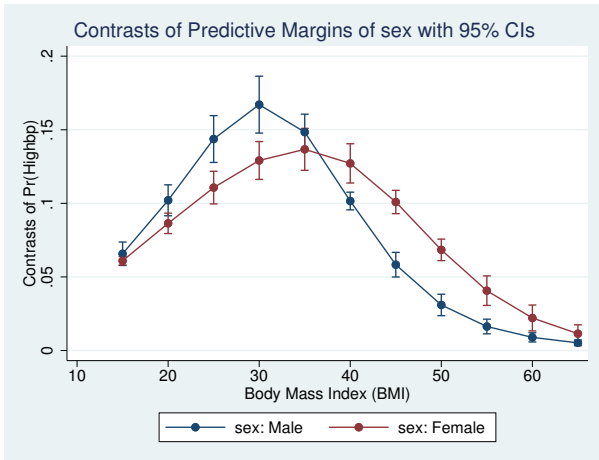
```
. logistic highbp sex##agegrp##c.bmi
. margins sex, at(bmi=(10(5)65))
. marginsplot, xlabel(10(10)60)
```

We can estimate the discrete effects by `sex` of `bmi` moving from 10 to 15, then from 15 to 20, . . . , and then from 60 to 65 by contrasting the levels of the `at()` groups using the reverse-adjacent contrast operator (`ar.`). We specify the operator within the `atcontrast()` suboption of the `contrast()` option. We need to specify one other option. By default, `margins, contrast` will apply a contrast to all variables in its *marginlist* when a contrast has been requested. In this case, we do not want to contrast across sexes but rather to contrast across the levels of BMI within each sex. To prevent `margins` from contrasting across the sexes, we specify the `marginswithin` option. Our `margins` command is

```
. margins sex, at(bmi=(10(5)65)) contrast(atcontrast(ar._at) marginswithin)
```

And we graph the results using `marginsplot`:

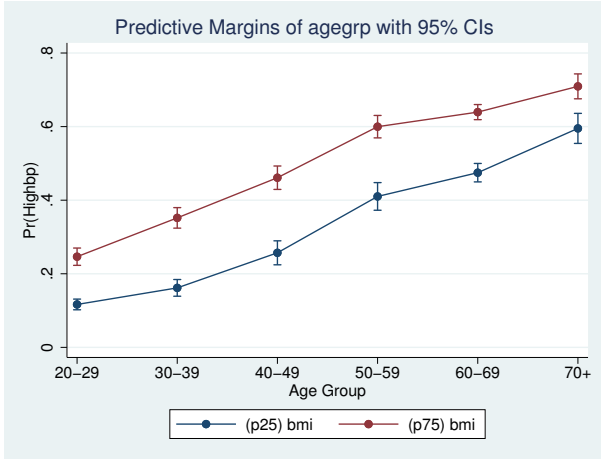
```
. marginsplot
  Variables that uniquely identify margins: bmi sex
```



The graph shows the contrasts (or if you prefer, discrete changes) in the probability of high blood pressure by sex as one increases BMI in 5-unit increments.

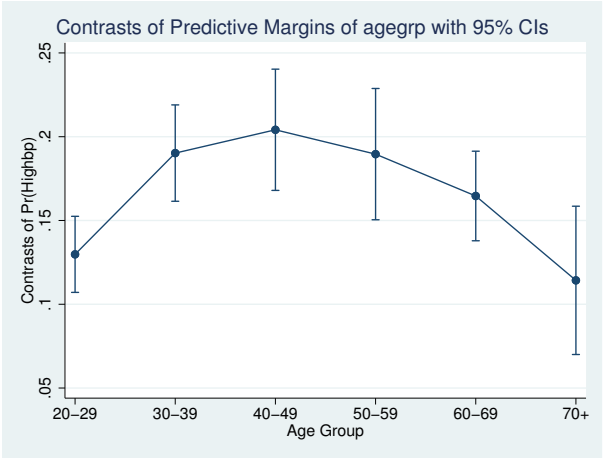
We can even estimate contrasts (discrete effects) across `at()` options. To start, let's compare the age-group profiles of the probability of high blood pressure for those in the 25th and 75th percentile of BMI.

```
. margins agegrp, at((p25) bmi) at((p75) bmi)
  (output omitted)
. marginsplot
  Variables that uniquely identify margins: agegrp _atopt
  Multiple at() options specified:
      _atoption=1: (p25) bmi
      _atoption=2: (p75) bmi
```



For each age group, people whose BMI is at the 75th percentile have a much higher probability of high blood pressure than those at the 25th percentile. What is that difference in probability and its CI? To contrast across the percentiles of BMI within age groups, we again specify a contrast operator on the `at()` groups using `atcontrast()`, and we also tell `margins` to perform that contrast within the levels of the *marginlist* by using the `marginswithin` option.

```
. margins agegrp, at((p25) bmi) at((p75) bmi)
> contrast(atcontrast(r._at) marginswithin)
  (output omitted)
. marginsplot
  Variables that uniquely identify margins: agegrp _atopt
  Multiple at() options specified:
      _atoption=1: (p25) bmi
      _atoption=2: (p75) bmi
```



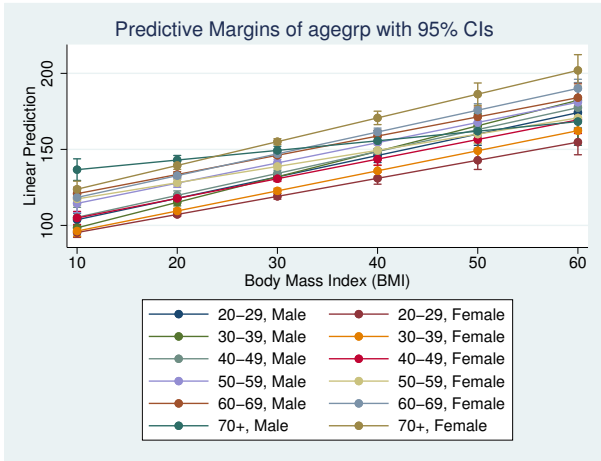
The differences in probability between 25th and 75th BMI percentiles are clearly significantly greater than 0. The differences appear to be smallest for those in the youngest and oldest age groups.

Controlling the graph's dimensions

Thus far, `marginsplot` has miraculously done almost exactly what we want in most cases. The things we want on the x axis have been there, the choice of plots has made sense, etc. Some of that luck sprang from the relatively simple analyses we were performing, and some was from careful specification of our `margins` command. Sometimes, we will not be so lucky.

Consider the following `regress`, `margins`, and `marginsplot` commands:

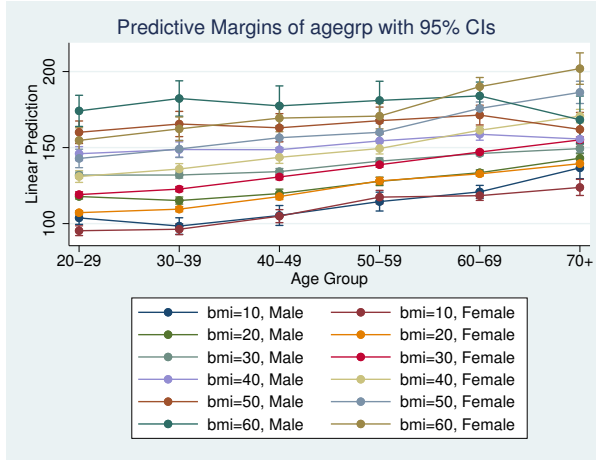
```
. regress bpsystol agegrp##sex##c.bmi
  (output omitted)
. margins agegrp, over(sex) at(bmi=(10(10)60))
  (output omitted)
. marginsplot
Variables that uniquely identify margins: agegrp _atopt
Multiple at() options specified:
    _atoption=1: (p25) bmi
    _atoption=2: (p75) bmi
```



By default, `marginsplot` places the levels of the first multilevel `at()` specification on the x axis, and then usually plots the levels of all remaining variables as connected lines. That is what we see in the graph above—`bmi`, the `at()` variable, is on the x axis, and each combination of `agegrp` and `sex` is plotted as a separate connected line. If there is no multilevel `at()` specification, then the first variable in *marginlist* becomes the x axis. There are many more rules, but it is usually best to simply type `marginsplot` and see what happens. If you do not like `marginsplot`'s choices, change them.

What if we wanted `agegrp` on the x axis instead of BMI? We tell `marginsplot` to make that change by specifying `agegrp` in the `xdimension()` option:

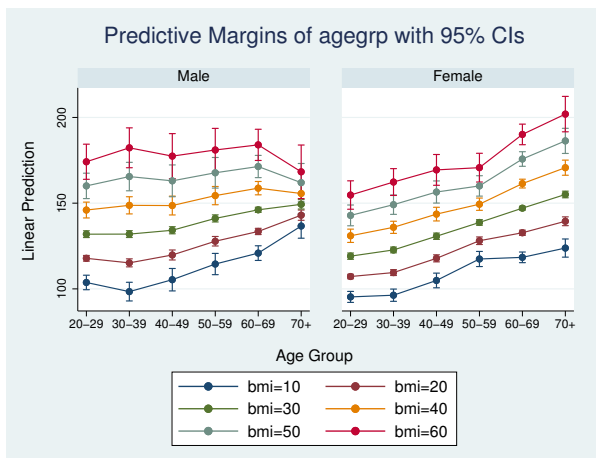
```
. marginsplot, xdimension(agegrp)
  Variables that uniquely identify margins: bmi agegrp sex
```



We have been suppressing the Results window output for `marginsplot`, but that output is helpful if we want to change how things are plotted. You may specify any variable used in your `margins` command in any of the dimension options—`xdimension()`, `plotdimension()`, `bydimension()`, and `graphdimension()`. (In fact, there are some pseudovariables that you may also specify in some cases; see *Addendum: Advanced uses of dimlist* for details.) `marginsplot` tries to help you narrow your choices by listing a set of variables that uniquely identify all your margins. You are not restricted to this list.

We have a different x axis and a different set of plots, but our graph is still busy and difficult to read. We can make it better by creating separate graph panels for each sex. We do that by adding a `bydimension()` option with `sex` as the argument.

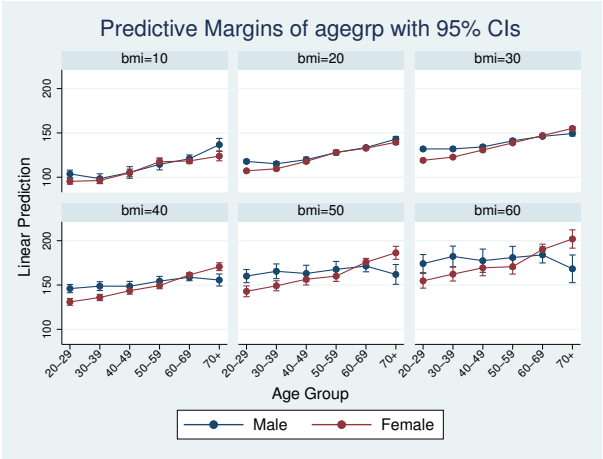
```
. marginsplot, xdimension(agegrp) bydimension(sex)
  Variables that uniquely identify margins: bmi agegrp sex
```



The patterns and the differences between males and females are now easier to see.

If our interest is in comparing males and females, we might even choose to create a separate panel for each level of BMI:

```
. marginsplot, xdimension(agegrp) bydimension(bmi) xlabel(, angle(45))
  Variables that uniquely identify margins: bmi agegrp sex
```



The x-axis labels did not fit, so we angled them.

We leave you to explore the use of the `graphdimension()` option. It is much like `bydimension()` but creates separate graphs rather than separate panels. Operationally, the `plotdimension()` option is rarely used. All variables not in the x dimension and not specified elsewhere become the plotted connected lines.

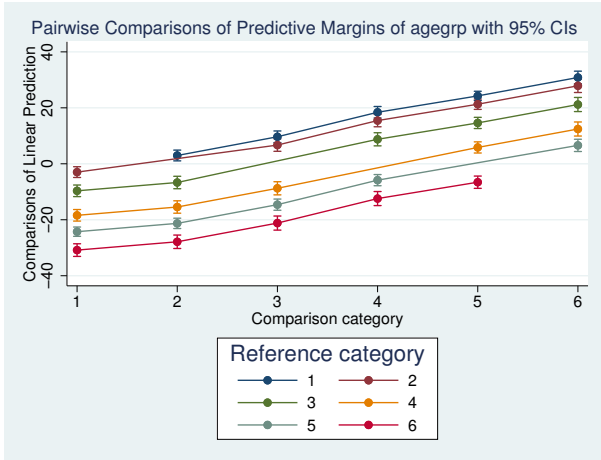
You will likely use the dimension options frequently. This is one of the rare cases where we recommend using the minimal abbreviations of the options—`x()` for `xdimension()`, `plot()` for `plotdimension()`, `by()` for `bydimension()`, and `graph()` for `graphdimension()`. The abbreviations are easy to read and just as meaningful as the full option names. The full names exist to reinforce the relationship between the dimension options.

Pairwise comparisons

`marginsplot` can graph the results of `margins, pwcompare`; see [R] **margins, pwcompare**. We return to one of our ANOVA examples. Here we request pairwise comparisons with the `pwcompare` option of `margins`, and we request Bonferroni-adjusted CIs with the `mcompare()` option:

```
. anova bpsystol agegrp##sex
  (output omitted)
. margins agegrp, pwcompare mcompare(bonferroni)
  (output omitted)
. marginsplot
  Variables that uniquely identify margins: _pw1 _pw0

      _pw enumerates all pairwise comparisons; _pw0 enumerates the reference
          categories; _pw1 enumerates the comparison categories.
```



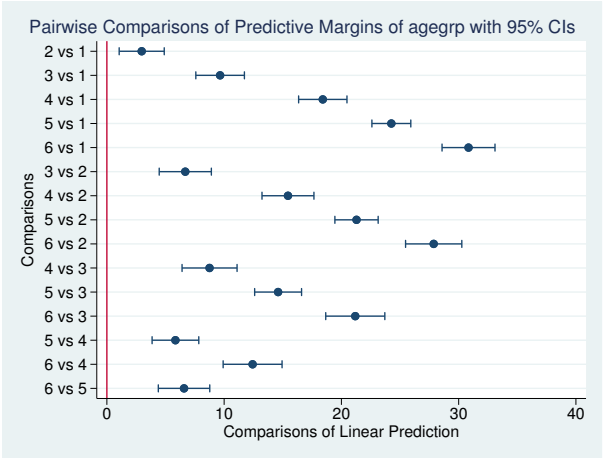
Each connected line plot in the graph represents a reference age-group category for the pairwise comparison. The ticks on the x axis represent comparison age-group categories. So, each plot is a profile for a reference category showing its comparison to each other category.

Horizontal is sometimes better

Another interesting way to graph pairwise comparisons is to simply plot each comparison and label the two categories being compared. This type of graph works better if it is oriented horizontally rather than vertically.

Continuing with the example above, we will switch the graph to horizontal. We will also make several changes to display the graph better. We specify that only `unique` comparisons be plotted. The graph above plotted both 1 versus 2 and 2 versus 1, which are the same comparison with opposite signs. We add a reference line at 0 because we are interested in comparisons that differ from 0. This graph looks better without the connecting lines, so we add the option `recast(scatter)`. We also reverse the y scale so that the smallest levels of age group appear at the top of the axis.

```
. marginsplot, horizontal unique xline(0) recast(scatter) yscale(reverse)
  Variables that uniquely identify margins: _pw1 _pw0
      _pw enumerates all pairwise comparisons; _pw0 enumerates the reference
          categories; _pw1 enumerates the comparison categories.
```



All the comparisons differ from 0, so all our age groups are statistically different from each other.

The `horizontal` option can be useful outside of pairwise comparisons. Profile plots are usually oriented vertically. However, when your covariates have long labels or there are many levels at which the margins are being evaluated, the graph may be easier to read when rendered horizontally.

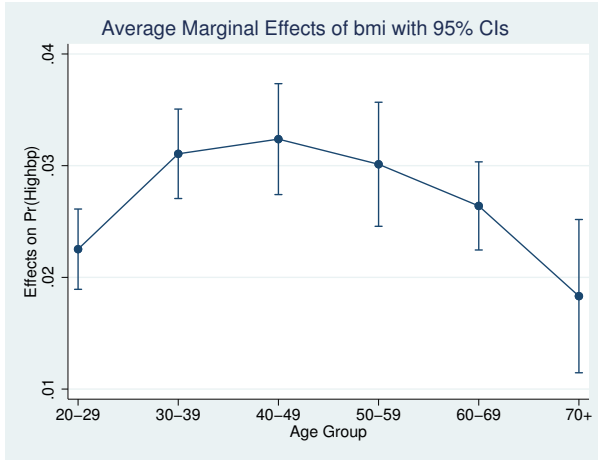
Marginal effects

We have seen how to graph discrete effects for factor variables and continuous variables by using contrasts, and optionally by using the `dydx()` option of `margins`: *Contrasts of margins—effects (discrete marginal effects)* and *Continuous covariates*. Let's now consider graphing instantaneous marginal effects for continuous covariates. Begin by refitting our logistic model of high blood pressure as a function of sex, age, and BMI:

```
. logistic highbp sex##agegrp##c.bmi
```

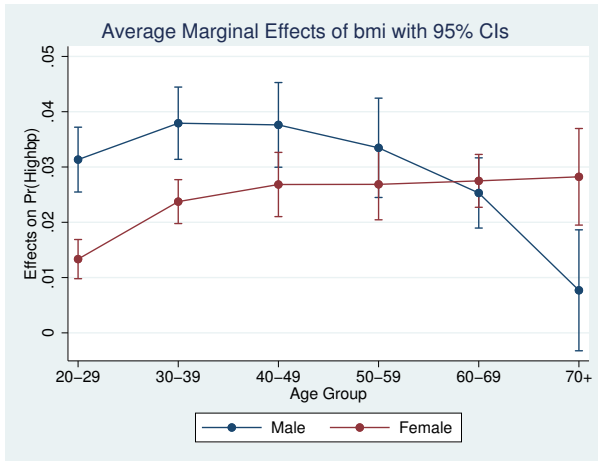
We estimate the average marginal effect of BMI on the probability of high blood pressure for each age group and then graph the results by typing

```
. margins agegrp, dydx(bmi)
(output omitted)
. marginsplot
Variables that uniquely identify margins: agegrp
```



These are the conditional expectations of the marginal effects treating everyone in the sample as though they were in each age group. We can estimate fully conditional marginal effects that do not depend on averaging over the sample by also margining on our one remaining covariate—`sex`.

```
. margins agegrp#sex, dydx(bmi)
(output omitted)
. marginsplot
Variables that uniquely identify margins: agegrp sex
```



The effect of BMI on the probability of high blood pressure looks to increase with age for females. The marginal effect is higher for males than females in the younger age groups but then decreases with age for males after the 40–49 age group.

You may want to test for differences in the marginal effect of BMI for males and females by contrasting across sexes within `agegrp`:

```
. margins r.sex@agegrp, dydx(bmi)
```

Plotting a subset of the results from margins

`marginsplot` plots all the margins produced by the preceding `margins` command. If you want a graph that does not include all the margins, then enter a `margins` command that produces a reduced set of margins. Obvious ways to reduce the number of margins include not specifying some factors or interactions in the *marginlist* of `margins`, not specifying some `at()` or `over()` options, or reducing the values specified in an `at()` option. A less obvious technique uses selection lists in factor operators to select specific sets of levels from factor variables specified in the *marginlist*.

Instead of typing

```
. margins agegrp
```

which will give you margins for all six age groups in our sample, type

```
. margins i(2/4).agegrp
```

which will give you only three margins—those for groups 2, 3, and 4. See **[U] 11.4.3.4 Selecting levels**.

Advanced usage

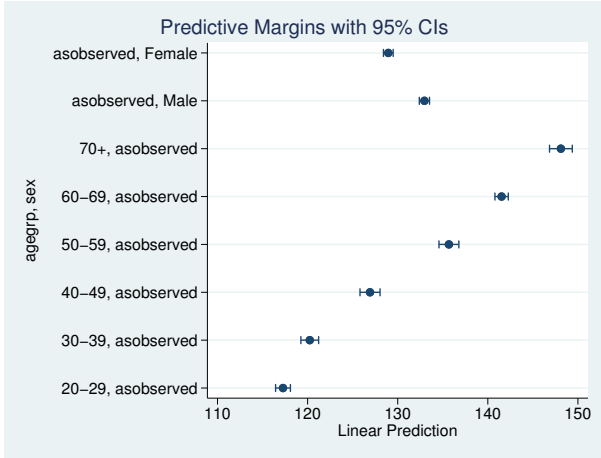
`margins` is incredibly flexible in the statistics it can estimate and in the grouping of those estimates. Many of the estimates that `margins` can produce do not make convincing graphs. `marginsplot` plots the results of any `margins` command, regardless of whether the resulting graph is easily interpreted. Here we demonstrate some options that can make complicated `margins` into graphs that are somewhat more useful than those produced by `marginsplot`'s defaults. Others may find truly useful applications for these approaches.

Plots with multiple terms

Margins plots are rarely interesting when you specify multiple terms on your `margins` command, for example, `margins a b`. Such plots often compare things that are not comparable. The defaults for `marginsplot` rarely produce useful plots with multiple terms. Perhaps the most interesting graph in such cases puts all the levels of all the terms together on the vertical axis and plots their margins on the horizontal axis. We do that by including the *marginlist* from `margins` in an `xdimension()` option on `marginsplot`. The long labels on such graphs look better with a horizontal orientation, and there is no need to connect the margin estimates, so we specify the `recast(scatter)` option.

Using one of our ANOVA examples from earlier,

```
. anova bpsystol agegrp##sex
```
(*output omitted*)
```
. margins agegrp sex
```
(*output omitted*)
```
. marginsplot, xdimension(agegrp sex) horizontal recast(scatter)
  Variables that uniquely identify margins: agegrp sex
```



The "asobserved" notations in the y-axis labels are informing us that, for example, when the margin for females is evaluated, the values of age group are taken as they are observed in the dataset. The margin is computed as an average over those values.

Plots with multiple at() options

Some disciplines like to compute margins at the means of other covariates in their model and others like to compute the response for each observation and then take the means of the response. These correspond to the `margins` options `at((mean) _all)` and `at((asobserved) _all)`. For responses that are linear functions of the coefficients, such as `predict` after `regress`, the two computations yield identical results. For responses that are nonlinear functions of the coefficients, the two computations estimate different things.

Using one of our logistic models of high blood pressure,

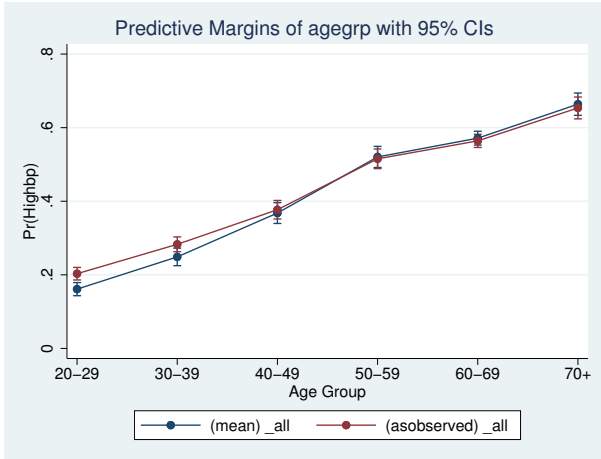
```
. logistic highbp sex##agegrp##c.bmi
```

and computing both sets of margins for each age group,

```
. margins agegrp, at((mean) _all) at((asobserved) _all)
```

we can use `marginsplot` to compare the approaches:

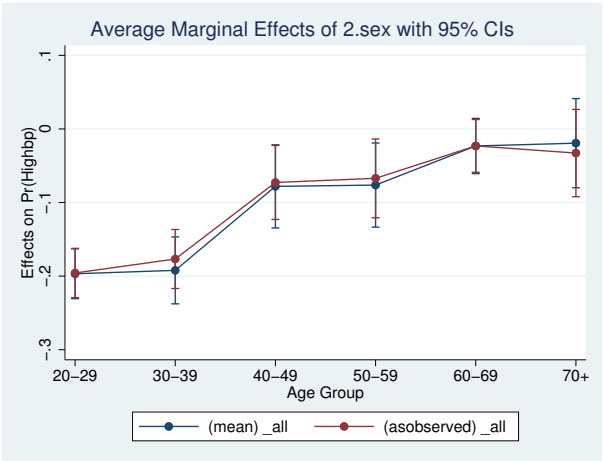
```
. marginsplot
Variables that uniquely identify margins: agegrp _atopt
Multiple at() options specified:
    _atoption=1: (mean)_all
    _atoption=2: (asobserved) _all
```



For the first three age groups, the probabilities of high blood pressure are lower at the means of `sex` and `bpi` than are the mean probabilities of high blood pressure averaged over the observed values of `sex` and `bpi`. The reverse is true for the last three age groups, although the values are very similar in these older age groups.

Such comparisons come up even more frequently when evaluating marginal effects. We can estimate the marginal effects of `sex` at each age group and graph the results by adding `dydx(sex)` to our `margins` command:

```
. margins agegrp, at((mean) _all) at((asobserved) _all) dydx(sex)
  (output omitted)
. marginsplot
  Variables that uniquely identify margins: agegrp _atopt
  Multiple at() options specified:
      _atoption=1: (mean)_all
      _atoption=2: (asobserved) _all
```

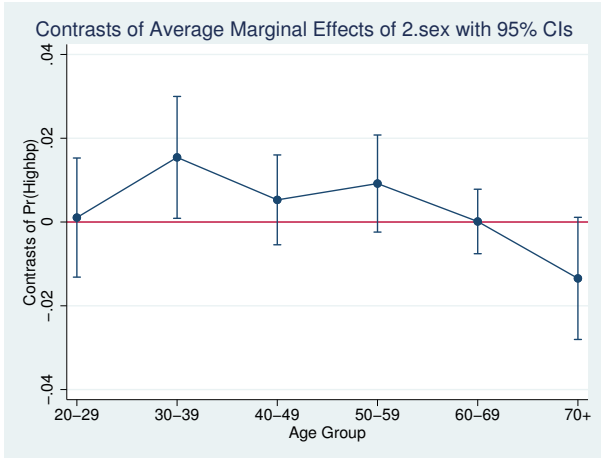


The average marginal effect is smaller for most age groups, but the CIs for both sets of estimates are wide. Can we tell the difference between the estimates? To answer that, we use the now-familiar tactic of taking the contrast of our estimated marginal-effects profiles. That means adding `contrast(atjoint marginswithin)` to our `margins` command. We will also add `mcompare(bonferroni)` to account for the fact that we will be comparing six contrasts.

```
. margins agegrp, at((mean) _all) at((asobserved) _all) dydx(sex)
> contrast(atjoint marginswithin) mcompare(bonferroni)
```

We will also add the familiar reference line at 0 to our graph of the contrasts.

```
. marginsplot, yline(0)
  Variables that uniquely identify margins: agegrp _atopt
  Multiple at() options specified:
      _atoption=1: (mean)_all
      _atoption=2: (asobserved) _all
```



While the difference in the estimates of marginal effects is not large, we can distinguish the estimates for the 30–39 and 70+ age groups.

The `at()` option of `margins` provides far more flexibility than demonstrated above. It can be used to evaluate a response or marginal effect at almost any point of interest or combinations of such points. See *Syntax of at()* in [R] **margins**.

Adding scatterplots of the data

We can add scatterplots of the observed data to our plots of the margins. The NHANES II dataset is too large for this to be interesting, so for this example, we will use `auto.dta`. We fit mileage on whether the care is foreign and on a quadratic in the weight of the car. We convert the weight into tons (U.S. definition) to improve the scaling, and we format the new `tons` variable to improve its labels on the graph. For our graph, we create separate variables for mileage of domestic and of foreign cars. We fit a fully interacted model so that the effect of weight on mileage can be different for foreign and for domestic cars.

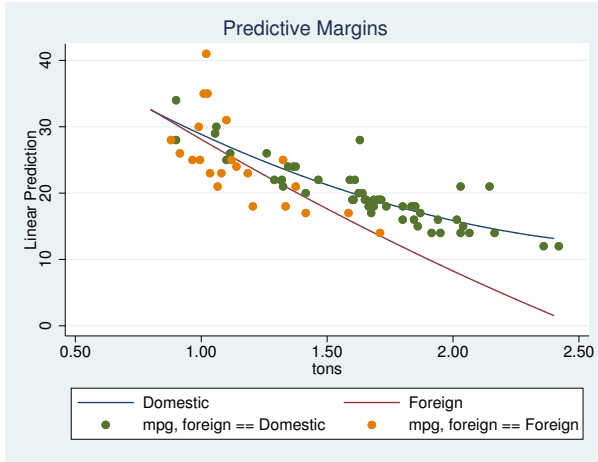
```
. use http://www.stata-press.com/data/r15/auto
. generate tons = weight/2000
. format tons %6.2f
. separate mpg, by(foreign)
. regress mpg foreign##c.tons##c.tons
```

We then estimate the margins over the range of tons, using the option `over(foreign)` to obtain separate estimates for foreign and domestic cars.

```
. margins, at(tons=(.8(.05)2.4)) over(foreign)
```

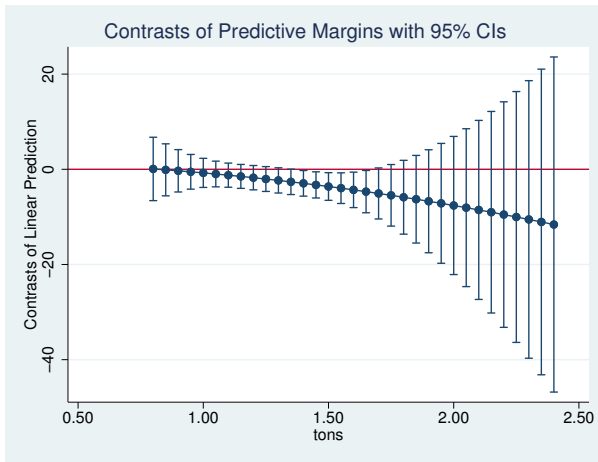
Adding scatterplots of mileage for domestic and foreign cars is easy. We insert into an `addplot()` option of `marginsplot` the same scatterplot syntax for `twoway` that we would type to produce a scatterplot of the data:

```
. marginsplot, addplot(scatter mpg0 tons || scatter mpg1 tons) recast(line) noci
  Variables that uniquely identify margins: tons foreign
```



Many will be surprised that the mileage profile is higher in 1978 for domestic (U.S. built) cars. Is the difference significant?

```
. margins, at(tons=(.8(.05)2.4)) over(r.for)
  (output omitted)
. marginsplot, yline(0)
  Variables that uniquely identify margins: tons
```



As we did earlier, we contrast the two profiles. We can discern some difference between the two profiles for midweight vehicles, but otherwise there is insufficient information to believe mileage differs across domestic and foreign cars.

Video examples

Profile plots and interaction plots, part 1: A single categorical variable

Profile plots and interaction plots, part 2: A single continuous variable

Profile plots and interaction plots, part 3: Interactions between categorical variables

Profile plots and interaction plots, part 4: Interactions of continuous and categorical variables

Profile plots and interaction plots, part 5: Interactions of two continuous variables

Addendum: Advanced uses of dimlist

dimlist specifies the dimensions from the immediately preceding `margins` command that are to be used for the `marginsplot`'s x axis, plots, subgraphs, and graphs. *dimlist* may contain:

dim	Description
varname	Any variable referenced in the preceding `margins` command.
`_equation`	If the estimation command being analyzed is multivariate and `margins` automatically produced estimates for more than one dependent-variable equation, then *dimlist* may contain `_equation` to enumerate those equations.
`_outcome`	If the estimation command being analyzed is ordinal and `margins` automatically produced estimates for more than one outcome level, then *dimlist* may contain `_outcome` to enumerate those outcomes.
`_predict`	If the preceding `margins` command included multiple `predict()` options, then *dimlist* may contain `_predict` to enumerate those `predict()` options.
`at(`*varname*`)`	If a variable is specified in both the *marginlist* or the `over()` option and in the `at()` option of `margins`, then the two uses can be distinguished in `marginsplot` by typing the `at()` variables as `at(`*varname*`)` in *dimlist*.
`_deriv`	If the preceding `margins` command included a `dydx()`, `eyex()`, `dyex()`, or `eydx()` option, *dimlist* may also contain `_deriv` to specify all the variables over which derivatives were taken.
`_term`	If the preceding `margins` command included multiple terms (for example, `margins a b`), then *dimlist* may contain `_term` to enumerate those terms.
`_atopt`	If the preceding `margins` command included multiple `at()` options, then *dimlist* may contain `_atopt` to enumerate those `at()` options.

When the `pairwise` option is specified on `margins`, you may specify dimensions that enumerate the pairwise comparisons.

`_pw`	enumerates all the pairwise comparisons
`_pw0`	enumerates the reference categories of the comparisons
`_pw1`	enumerates the comparison categories of the comparisons

Acknowledgments

We thank Philip B. Ender (retired) of UCLA Academic Technology Services for his programs that demonstrated what could be done in this area. We also thank Michael N. Mitchell, author of the Stata Press books *Data Management Using Stata: A Practical Handbook*, *Interpreting and Visualizing Regression Models Using Stata*, *Stata for the Behavioral Sciences*, and *A Visual Guide to Stata Graphics*, for his generous advice and comprehensive insight into the application of margins and their plots.

References

Jann, B. 2014. Plotting regression coefficients and other estimates. *Stata Journal* 14: 708–737.

Lindsey, C. 2016. Estimating covariate effects after gmm. *The Stata Blog: Not Elsewhere Classified*. http://blog.stata.com/2016/10/04/estimating-covariate-effects-after-gmm/.

McDowell, A., A. Engel, J. T. Massey, and K. Maurer. 1981. Plan and operation of the Second National Health and Nutrition Examination Survey, 1976–1980. *Vital and Health Statistics* 1(15): 1–144.

Mitchell, M. N. 2012. *Interpreting and Visualizing Regression Models Using Stata*. College Station, TX: Stata Press.

——. 2015. *Stata for the Behavioral Sciences*. College Station, TX: Stata Press.

Pinzon, E. 2016. Effects of nonlinear models with interactions of discrete and continuous variables: Estimating, graphing, and interpreting. *The Stata Blog: Not Elsewhere Classified*. http://blog.stata.com/2016/07/12/effects-for-nonlinear-models-with-interactions-of-discrete-and-continuous-variables-estimating-graphing-and-interpreting/.

Royston, P. 2013. marginscontplot: Plotting the marginal effects of continuous predictors. *Stata Journal* 13: 510–527.

Williams, R. 2012. Using the margins command to estimate and interpret adjusted predictions and marginal effects. *Stata Journal* 12: 308–331.

Also see

[R] **margins** — Marginal means, predictive margins, and marginal effects

[R] **margins, contrast** — Contrasts of margins

[R] **margins, pwcompare** — Pairwise comparisons of margins

[R] **margins postestimation** — Postestimation tools for margins