Stata Factor Variables

1. Stata's "factor variable" notation is ideal for categorical variables and interaction terms. For example, when Stata sees the prefix `i.` before a categorical variable that takes on *k* values, it knows to expand the categorical variable into *k-1* dummy variables. It does this temporarily for whatever command is using the factor variable. The dummies are "virtual" and not added to your dataset.

   For example, suppose your variable *boro* takes on five values for the NYC boroughs (coded 1-5: Brooklyn, Manhattan, Staten Island, Queens, and the Bronx). Factor variable notation can be used to get means for the virtual *boro* dummy variables:

   ```
   summ i.boro
   ```

   The resulting output will be:

   ```
       Variable |        Obs        Mean    Std. Dev.       Min        Max
   -------------+---------------------------------------------------------
          boro2 |
             2  |        437    .2448513    .4304918          0          1
             3  |        437    .1830664    .3871642          0          1
             4  |        437    .0228833    .1497028          0          1
             5  |        437    .2723112    .4456594          0          1
   ```

   Note missing values in the original data carry forward as missing values in the virtual dummies.

2. Factor variable notation can only be used with non-negative integer values. If your variable *boro* were a string (e.g., K, M, R, Q, X), you can first `encode` it before using factor notation:

   ```
   encode boro, gen(boro2)
   ```

   The results will be a numeric variable *boro2* with values 1-5. You may want to apply value labels to these numeric values if want them to appear in your output later.

3. By default, Stata creates *k-1* virtual dummy variables for categorical variables. One category is excluded. (Notice in the output above that only four boroughs are shown, 2-5. Borough 1 was omitted). You can control which group—if any—is excluded. A few options are:

`summ ibn.boro`	No category (or base level) excluded
`summ ib2.boro`	Category two is the excluded base level
`summ ib(freq).boro`	The most frequent category is the excluded base level
`summ ib(first).boro`	The first ordered category is the excluded base level

4. Factor variables are very useful for interaction terms. For example, the # notation will produce a *two-way interaction* between the variables *boro* and *sex*—that is, all combinations of values that *boro* and *sex* can take on:

```
summ i.boro#i.sex
```

There are 10 possible combinations in this interaction: K-Male, K-Female, M-Male, M-Female, and so on. Again, Stata will omit one dummy variable unless you specify otherwise with `ibn`. Note it is not necessary to include the `i.` notation here—Stata will assume with the two-way interaction that the variables are categorical.

5. The `##` operator produces interaction terms and main effects (that is, factorial interactions). Again, Stata will omit one dummy variable unless you specify otherwise with `ibn`.

```
summ i.boro##i.sex
```

6. Factor notation can be used for higher level-interactions. For example, the following example will produce a three-way interaction:

```
summ i.boro#i.sex#i.ell
```

7. The prefix `c.` before a variable name tells Stata that the variable is continuous. This can be used to create higher-order polynomial terms, and interactions between continuous and categorical variables. For example, the following command will include the continuous variable *age* and its square in the `regress` command:

```
regress hrwage age c.age#c.age
```

This command will include *age, female*, and their interaction in a regression:

```
regress hrwage i.female##c.age
```

8. There are many other things one can do with factor variables. A few examples follow. (Type "help factor variables" for more ideas).

`summ i2.boro`	Uses a virtual dummy for boro==2
`summ io(3 4).boro`	Uses virtual dummies for boro other than 3 and 4

9. You can use the `fvset` command to declare base settings that you intend to use repeatedly with the same variables. For example:

`fvset base 3` *varlist*	Use 3 as the base for each var in *varlist*
`fvset base none` *varlist*	No base category for each var in *varlist*
`fvset base first` *varlist*	Use first category as the base (default)
`fvset report [`*varlist*`]`	View current factor variable base settings
`fvset clear [`*varlist*`]`	Clear current factor variable base settings

10. Aside from convenience, the real power of factor variables comes in when combined with post-estimation commands like `margins`.

Last updated: September 18, 2021