# Title

stata.com

**mi impute regress —** Impute using linear regression



## Description

`mi impute regress` fills in missing values of a continuous variable using the Gaussian normal regression imputation method. You can perform separate imputations on different subsets of the data by specifying the `by()` option. You can also account for analytic, frequency, importance, and sampling weights.

## Menu

Statistics > Multiple imputation

## Syntax

`mi impute regress` *ivar* [ *indepvars* ] [ *if* ] [ *weight* ] [ , *impute_options* *options* ]

| *impute_options* | Description |
|---|---|
| Main | |
| * `add(#)` | specify number of imputations to add; required when no imputations exist |
| * `replace` | replace imputed values in existing imputations |
| `rseed(#)` | specify random-number seed |
| `double` | store imputed values in double precision; the default is to store them as `float` |
| `by(`*varlist*[ , *byopts* ]`)` | impute separately on each group formed by *varlist* |
| Reporting | |
| `dots` | display dots as imputations are performed |
| `noisily` | display intermediate output |
| `nolegend` | suppress all table legends |
| Advanced | |
| `force` | proceed with imputation, even when missing imputed values are encountered |
| `noupdate` | do not perform `mi update`; see [MI] **noupdate option** |

*`add(#)` is required when no imputations exist; `add(#)` or `replace` is required if imputations exist.
`noupdate` does not appear in the dialog box.

| *options* | Description |
|---|---|
| Main | |
| noconstant | suppress constant term |
| conditional(*if*) | perform conditional imputation |
| bootstrap | estimate model parameters using sampling with replacement |

You must mi set your data before using mi impute regress; see [MI] **mi set**.

You must mi register *ivar* as imputed before using mi impute regress; see [MI] **mi set**.

*indepvars* may contain factor variables; see [U] **11.4.3 Factor variables**.

collect is allowed; see [U] **11.1.10 Prefix commands**.

aweights, fweights, iweights, and pweights are allowed; see [U] **11.1.6 weight**.

# Options

## Main

noconstant; see [R] **Estimation options**.

add(), replace, rseed(), double, by(); see [MI] **mi impute**.

conditional(*if*) specifies that the imputation variable be imputed conditionally on observations satisfying *exp*; see [U] **11.1.3 if exp**. That is, missing values in a conditional sample, the sample identified by the *exp* expression, are imputed based only on data in that conditional sample. Missing values outside the conditional sample are replaced with a conditional constant, the value of the imputation variable in observations outside the conditional sample. As such, the imputation variable is required to be constant outside the conditional sample. Also, if any conditioning variables (variables involved in the conditional specification if *exp*) contain soft missing values (.), their missing values must be nested within missing values of the imputation variables. See *Conditional imputation* under *Remarks and examples* in [MI] **mi impute**.

bootstrap specifies that posterior estimates of model parameters be obtained using sampling with replacement; that is, posterior estimates are estimated from a bootstrap sample. The default is to sample the estimates from the posterior distribution of model parameters or from the large-sample normal approximation of the posterior distribution. This option is useful when asymptotic normality of parameter estimates is suspect.

## Reporting

dots, noisily, nolegend; see [MI] **mi impute**. noisily specifies that the output from a linear regression fit to the observed data be displayed. nolegend suppresses all legends that appear before the imputation table. Such legends include a legend about conditional imputation that appears when the conditional() option is specified and group legends that may appear when the by() option is specified.

## Advanced

force; see [MI] **mi impute**.

The following option is available with mi impute but is not shown in the dialog box:

noupdate; see [MI] **noupdate option**.

# Remarks and examples

stata.com

Remarks are presented under the following headings:



See [MI] **mi impute** for a general description and details about options common to all imputation methods, *impute_options*. Also see [MI] **Workflow** for general advice on working with `mi`.

## Univariate imputation using linear regression

When a continuous variable contains missing values, a linear regression imputation method (or predictive mean matching; see [MI] **mi impute pmm**) can be used to fill in missing values (Rubin 1987; Schenker and Taylor 1996). The linear regression method is a fully parametric imputation method that relies on the normality of the model. Thus the imputation variable may need to be transformed from the original scale to meet the normality assumption prior to using `mi impute regress`.

The linear regression method is perhaps the most popular method for imputing quantitative variables. It is superior to other imputation methods when the underlying normal model holds. However, it can be more sensitive to violations of this assumption than other nonparametric and partially parametric imputation methods, such as predictive mean matching. For example, Schenker and Taylor (1996) studied the sensitivity of the regression method to the misspecification of the regression function and error distribution. They found that this method still performs well in the presence of heteroskedasticity and when the error distribution is heavier-tailed than the normal. However, it resulted in increased bias and variances under a misspecified regression function.

## Using mi impute regress

Recall the heart attack data from *Univariate imputation* of [MI] **mi impute**. We wish to fit a logistic regression of `attack` on some predictors, one of which (`bmi`) has missing values. To avoid losing information contained in complete observations of the other predictors, we impute `bmi`.

The distribution of BMI is slightly skewed to the right, so we choose to fill in missing values of BMI on a log-transformed scale here. To do that, we need to create a new variable, `lnbmi`, containing the log of `bmi` and impute it:

```
. use https://www.stata-press.com/data/r17/mheart0
(Fictional heart attack data; BMI missing)
. generate lnbmi = ln(bmi)
(22 missing values generated)
. mi set mlong
. mi register imputed lnbmi
(22 m=0 obs now marked as incomplete)
```

Following the steps in *Imputing transformations of incomplete variables* of [MI] **mi impute**, we create the imputed variable `lnbmi` containing the log of `bmi` and register it as imputed. We omitted the step of eliminating possible ineligible missing values in `lnbmi` because `bmi` ranges from 17 to 38 and we do not anticipate any extra (algebraic) missing from the operation `ln(bmi)`.

We now use `mi impute` to impute missing values of `lnbmi`. We create 20 imputations and specify a random-number seed for reproducibility:

```
. mi impute regress lnbmi attack smokes age hsgrad female, add(20) rseed(2232)
Univariate imputation                       Imputations =       20
Linear regression                                 added =       20
Imputed: m=1 through m=20                       updated =        0
```

| | Observations per $m$ | | | |
|---|---|---|---|---|
| Variable | Complete | Incomplete | Imputed | Total |
| lnbmi | 132 | 22 | 22 | 154 |

```
(Complete + Incomplete = Total; Imputed is the minimum across m
 of the number of filled-in observations.)
```

From the output, all 22 incomplete values of `lnbmi` are imputed.

We want to use BMI in its original scale in the analysis. To do that, we need to replace `bmi` with exponentiated `lnbmi`. Because `bmi` now is a function of the imputed variable, it becomes a passive variable:

```
. mi register passive bmi
. quietly mi passive: replace bmi = exp(lnbmi)
```

Finally, we fit the logistic regression:

```
. mi estimate, dots: logit attack smokes age bmi hsgrad female
Imputations (20):
  .........10.........20 done
Multiple-imputation estimates                   Imputations       =         20
Logistic regression                             Number of obs     =        154
                                                Average RVI       =     0.0636
                                                Largest FMI       =     0.2619
DF adjustment:   Large sample                   DF:     min       =     288.05
                                                        avg       = 121,496.74
                                                        max       = 215,505.49
Model F test:       Equal FMI                   F(   5,18140.5)   =       3.51
Within VCE type:          OIM                   Prob > F          =     0.0036
```

| attack | Coefficient | Std. err. | t | P>\|t\| | [95% conf. | interval] |
|---|---|---|---|---|---|---|
| smokes | 1.222696 | .3606843 | 3.39 | 0.001 | .5157605 | 1.929631 |
| age | .0358906 | .0154728 | 2.32 | 0.020 | .0055643 | .0662169 |
| bmi | .1092684 | .0522189 | 2.09 | 0.037 | .0064894 | .2120473 |
| hsgrad | .1733616 | .405482 | 0.43 | 0.669 | -.621373 | .9680961 |
| female | -.0987609 | .4186695 | -0.24 | 0.814 | -.9193435 | .7218218 |
| _cons | -5.625106 | 1.791905 | -3.14 | 0.002 | -9.143989 | -2.106223 |

We obtain results comparable with those from [MI] **Intro substantive**.

## Video example

Multiple imputation: Setup, imputation, estimation—regression imputation

## Stored results

`mi impute regress` stores the following in `r()`:

Scalars

| | |
|---|---|
| `r(M)` | total number of imputations |
| `r(M_add)` | number of added imputations |
| `r(M_update)` | number of updated imputations |
| `r(k_ivars)` | number of imputed variables (always `1`) |
| `r(N_g)` | number of imputed groups (`1` if `by()` is not specified) |

Macros

| | |
|---|---|
| `r(method)` | name of imputation method (`regress`) |
| `r(ivars)` | names of imputation variables |
| `r(rngstate)` | random-number state used |
| `r(by)` | names of variables specified within `by()` |

Matrices

| | |
|---|---|
| `r(N)` | number of observations in imputation sample in each group |
| `r(N_complete)` | number of complete observations in imputation sample in each group |
| `r(N_incomplete)` | number of incomplete observations in imputation sample in each group |
| `r(N_imputed)` | number of imputed observations in imputation sample in each group |

## Methods and formulas

Consider a univariate variable $\mathbf{x} = (x_1, x_2, \ldots, x_n)'$ that follows a normal linear regression model

$$x_i|\mathbf{z}_i \sim N(\mathbf{z}_i'\boldsymbol{\beta}, \sigma^2) \tag{1}$$

where $\mathbf{z}_i = (z_{i1}, z_{i2}, \ldots, z_{iq})'$ records values of predictors of $\mathbf{x}$ for observation $i$, $\boldsymbol{\beta}$ is the $q \times 1$ vector of unknown regression coefficients, and $\sigma^2$ is the unknown scalar variance. (Note that when a constant is included in the model—the default—$z_{i1} = 1,\ i = 1, \ldots, n$.)

$\mathbf{x}$ contains missing values that are to be filled in. Consider the partition of $\mathbf{x} = (\mathbf{x}_o', \mathbf{x}_m')$ into $n_0 \times 1$ and $n_1 \times 1$ vectors containing the complete and the incomplete observations. Consider a similar partition of $\mathbf{Z} = (\mathbf{Z}_o, \mathbf{Z}_m)$ into $n_0 \times q$ and $n_1 \times q$ submatrices.

`mi impute regress` follows the steps below to fill in $\mathbf{x}_m$ (for simplicity, we omit the conditioning on the observed data in what follows):

1. Fit a regression model (1) to the observed data $(\mathbf{x}_o, \mathbf{Z}_o)$ to obtain estimates $\widehat{\boldsymbol{\beta}}$ and $\widehat{\sigma}^2$ of the model parameters.
2. Simulate new parameters $\boldsymbol{\beta}_\star$ and $\sigma^2_\star$ from their joint posterior distribution under the conventional noninformative improper prior $\Pr(\boldsymbol{\beta}, \sigma^2) \propto 1/\sigma^2$. This is done in two steps:

$$\begin{aligned} \sigma^2_\star &\sim \widehat{\sigma}^2(n_0 - q)/\chi^2_{n_0-q} \\ \boldsymbol{\beta}_\star|\sigma^2_\star &\sim N\left\{\widehat{\boldsymbol{\beta}},\ \sigma^2_\star(\mathbf{Z}_o'\mathbf{Z}_o)^{-1}\right\} \end{aligned}$$

3. Obtain one set of imputed values, $\mathbf{x}_m^1$, by simulating from $N(\mathbf{Z}_m\boldsymbol{\beta}_\star,\ \sigma^2_\star I_{n_1 \times n_1})$.
4. Repeat steps 2 and 3 to obtain $M$ sets of imputed values, $\mathbf{x}_m^1, \mathbf{x}_m^2, \ldots, \mathbf{x}_m^M$.

Steps 2 and 3 above correspond to simulating from the posterior predictive distribution of the missing data $\Pr(\mathbf{x}_m|\mathbf{x}_o, \mathbf{Z}_o)$ (for example, see Gelman et al. [2014, 354–357]).

If weights are specified, a weighted linear regression model is fit to the observed data in step 1 (see [R] **regress** for details). Also, in the case of `aweights`, $\sigma^2_\star I_{n_1\times n_1}$ is replaced with $\sigma^2_\star \mathbf{W}^{-1}_{n_1\times n_1}$ in step 3, where $\mathbf{W} = \text{diag}(w_i)$ and $w_i$ is the analytic weight for observation $i$.

## References

Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. 2014. *Bayesian Data Analysis*. 3rd ed. Boca Raton, FL: Chapman & Hall/CRC.

Rubin, D. B. 1987. *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley.

Schenker, N., and J. M. G. Taylor. 1996. Partially parametric techniques for multiple imputation. *Computational Statistics & Data Analysis* 22: 425–446. https://doi.org/10.1016/0167-9473(95)00057-7.

## Also see

[MI] **mi impute** — Impute missing values

[MI] **mi impute intreg** — Impute using interval regression

[MI] **mi impute pmm** — Impute using predictive mean matching

[MI] **mi impute truncreg** — Impute using truncated regression

[MI] **mi estimate** — Estimation using multiple imputations

[MI] **Intro** — Introduction to mi

[MI] **Intro substantive** — Introduction to multiple-imputation analysis