

mi impute pmm — Impute using predictive mean matching

[Syntax](#)[Remarks and examples](#)[Also see](#)[Menu](#)[Stored results](#)[Description](#)[Methods and formulas](#)[Options](#)[References](#)

Syntax

```
mi impute pmm 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 (<i>varlist</i> [, <i>byopts</i>])	impute separately on each group formed by <i>varlist</i>

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
knn (#)	specify # of closest observations (nearest neighbors) to draw from; default is knn (1)
conditional (<i>if</i>)	perform conditional imputation
bootstrap	estimate model parameters using sampling with replacement

You must **mi set** your data before using **mi impute pmm**; see [\[MI\] mi set](#).

You must **mi register** *ivar* as imputed before using **mi impute pmm**; see [\[MI\] mi set](#).

indepvars may contain factor variables; see [\[U\] 11.4.3 Factor variables](#).

awweights, **fweights**, **iweights**, and **pweights** are allowed; see [\[U\] 11.1.6 weight](#).

Menu

Statistics > Multiple imputation

Description

`mi impute pmm` fills in missing values of a continuous variable by using the predictive mean matching 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.

Options

Main

`noconstant`; see [\[R\] estimation options](#).

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

`knn(#)` specifies the number of closest observations (nearest neighbors) from which to draw imputed values. The default is to replace a missing value with the “closest” observation, `knn(1)`. The closeness is determined based on the absolute difference between the linear prediction for the missing value and that for the complete values. The closest observation is the observation with the smallest difference. This option regulates the correlation among multiple imputations that affects the bias and the variability of the resulting multiple-imputation point estimates; see [Remarks and examples](#) for details.

Recent simulation studies demonstrate that using one nearest neighbor performs poorly in many of the considered scenarios. In general, the optimal number of nearest neighbors varies from one application to another; see [Morris, White, and Royston \(2014\)](#) for details.

`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 the 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:

[Univariate imputation using predictive mean matching](#)

[Using mi impute pmm](#)

[Video example](#)

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 predictive mean matching

Either predictive mean matching (`pmm`) or normal linear regression (`regress`) imputation methods can be used to fill in missing values of a continuous variable (Rubin 1987; Schenker and Taylor 1996). Predictive mean matching may be preferable to linear regression when the normality of the underlying model is suspect.

Predictive mean matching (PMM) is a partially parametric method that matches the missing value to the observed value with the closest predicted mean (or linear prediction). It was introduced by Little (1988) based on Rubin's (1986) ideas applied to statistical file matching. PMM combines the standard linear regression and the nearest-neighbor imputation approaches. It uses the normal linear regression to obtain linear predictions. It then uses the linear prediction as a distance measure to form the set of nearest neighbors (possible donors) consisting of the complete values. Finally, it randomly draws an imputed value from this set. By drawing from the observed data, PMM preserves the distribution of the observed values in the missing part of the data, which makes it more robust than the fully parametric linear regression approach.

With PMM, you need to decide how many nearest neighbors to include in the set of possible donors. `mi impute pmm` defaults to one nearest neighbor, `knn(1)`. Recent simulation studies demonstrate that using one nearest neighbor performs poorly in many of the considered scenarios (Morris, White, and Royston 2014), so you may need to include more depending on your data. The number of nearest neighbors affects the correlation among imputations—the smaller the number, the higher the correlation. High correlation in turn increases the variability of the MI point estimates. Including too many possible donors may result in increased bias of the MI point estimates. Thus the number of nearest neighbors regulates the tradeoff between the bias and the variance of the point estimators in repeated sampling. The literature does not provide a definitive recommendation on how to choose this number in practice; see Schenker and Taylor (1996) for some insight into this issue.

Using mi impute pmm

Recall the heart attack data from *Univariate imputation* in [\[MI\] 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`.

We showed one way of imputing `bmi` in [\[MI\] mi impute regress](#). Suppose, however, that we want to restrict the imputed values of `bmi` to be within the range observed for `bmi`. We can use the PMM imputation method to restrict the values. This method may also be preferable to the regression imputation of `bmi` because the distribution of `bmi` is slightly skewed.

```
. use http://www.stata-press.com/data/r13/mheart0
(Fictional heart attack data; bmi missing)
. mi set mlong
. mi register imputed bmi
(22 m=0 obs. now marked as incomplete)
. mi impute pmm bmi attack smokes age hsgrad female, add(20)
Univariate imputation           Imputations =      20
Predictive mean matching         added =      20
Imputed: m=1 through m=20       updated =       0
                                   Nearest neighbors =      1
```

Variable	Observations per <i>m</i>			Total
	Complete	Incomplete	Imputed	
bmi	132	22	22	154

(complete + incomplete = total; imputed is the minimum across *m* of the number of filled-in observations.)

By default, `mi impute pmm` uses one nearest neighbor to draw from. That is, it replaces missing values with an observed value whose linear prediction is the closest to that of the missing value. This default is arbitrary and may perform poorly depending on your data; see [Morris, White, and Royston \(2014\)](#). Using only one nearest neighbor may result in high variability of the MI estimates. You can increase the number of nearest neighbors from which the imputed value is drawn by specifying the `knn()` option. For example, we use 5 below:

```
. mi impute pmm bmi attack smokes age hsgrad female, replace knn(5)
Univariate imputation           Imputations =      20
Predictive mean matching         added =       0
Imputed: m=1 through m=20       updated =      20
                                   Nearest neighbors =       5
```

Variable	Observations per <i>m</i>			Total
	Complete	Incomplete	Imputed	
bmi	132	22	22	154

(complete + incomplete = total; imputed is the minimum across *m* of the number of filled-in observations.)

You can now refit the logistic model and examine the effect of using more neighbors:

```
. mi estimate: logit attack smokes age bmi hsgrad female
(output omitted)
```

See [\[MI\] mi impute](#), [\[MI\] mi impute regress](#), and [\[MI\] mi estimate](#) for more details.

Video example

Multiple imputation, part 2: Imputing a single continuous variable with predictive mean matching

Stored results

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

Scalars

<code>r(M)</code>	total number of imputations
<code>r(M_add)</code>	number of added imputations
<code>r(M_update)</code>	number of updated imputations
<code>r(knn)</code>	number of k nearest neighbors
<code>r(k_ivars)</code>	number of imputed variables (always 1)
<code>r(N_g)</code>	number of imputed groups (1 if <code>by()</code> is not specified)

Macros

<code>r(method)</code>	name of imputation method (<code>pmm</code>)
<code>r(ivars)</code>	names of imputation variables
<code>r(rseed)</code>	random-number seed
<code>r(by)</code>	names of variables specified within <code>by()</code>

Matrices

<code>r(N)</code>	number of observations in imputation sample in each group
<code>r(N_complete)</code>	number of complete observations in imputation sample in each group
<code>r(N_incomplete)</code>	number of incomplete observations in imputation sample in each group
<code>r(N_imputed)</code>	number of imputed observations in imputation sample in each group

Methods and formulas

`mi impute pmm` follows the steps as described in [Methods and formulas](#) of [MI] [mi impute regress](#) with the exception of step 3.

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

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

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

\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 pmm` 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 $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ of the model parameters.
2. Simulate new parameters $\boldsymbol{\beta}_*$ and σ_*^2 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:

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

3. Generate the imputed values, \mathbf{x}_m^1 , as follows. Let \hat{x}_i be the linear prediction of \mathbf{x} based on predictors \mathbf{Z} for observation i . Then for any missing observation i of \mathbf{x} , $x_i = x_{j_{\min}}$, where j_{\min} is randomly drawn from the set of indices $\{i_1, i_2, \dots, i_k\}$ corresponding to the first k minimums determined based on the absolute differences between the linear prediction for incomplete observation i and linear predictions for all complete observations, $|\hat{x}_i - \hat{x}_j|$, $j \in \text{obs}$. For example, if $k = 1$ (the default), j_{\min} is determined based on $|\hat{x}_i - \hat{x}_{j_{\min}}| = \min_{j \in \text{obs}} |\hat{x}_i - \hat{x}_j|$.
4. Repeat steps 2 and 3 to obtain M sets of imputed values, $\mathbf{x}_m^1, \mathbf{x}_m^2, \dots, \mathbf{x}_m^M$.

If weights are specified, a weighted linear regression model is fit to the observed data in step 1 (see [R] [regress](#) for details).

References

- Little, R. J. A. 1988. Missing-data adjustments in large surveys. *Journal of Business and Economic Statistics* 6: 287–296.
- Morris, T. P., I. R. White, and P. Royston. 2014. Tuning multiple imputation by predictive mean matching and local residual draws. *BMC Medical Research Methodology* 14: 75.
- Rubin, D. B. 1986. Statistical matching using file concatenation with adjusted weights and multiple imputations. *Journal of Business and Economic Statistics* 4: 87–94.
- . 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.

Also see

- [MI] [mi impute](#) — Impute missing values
- [MI] [mi impute intreg](#) — Impute using interval regression
- [MI] [mi impute regress](#) — Impute using linear regression
- [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