Supporting Information for "Missing data in ecology: Some synthesis, clarification, and recommendations"

Michael Dumelle[a,*], Rob Trangucci[b], Amanda M. Nahlik[a], Anthony R. Olsen[a], Kathryn M. Irvine[c], Karen Blocksom[a], Jay M. Ver Hoef[d], Claudio Fuentes[b]

[a]*United States Environmental Protection Agency, Office of Research and Development, 200 SW 35th St, Corvallis, OR, USA.*
[b]*Department of Statistics, Oregon State University, Corvallis, OR, USA*
[c]*United States Geological Survey, 2327 University Way, Suite 2, Bozeman, MT, USA.*
[d]*Marine Mammal Laboratory, Alaska Fisheries Science Center, NOAA Fisheries, Seattle, WA, USA*

[*]Corresponding Author; An author biosketch is provided in Appendix S1
Email address: Dumelle.Michael@epa.gov; michael.dumelle@oregonstate.edu (Michael Dumelle)

Appendix S1 Biosketch

Michael Dumelle is a statistician for the United States Environmental Protection Agency (USEPA) from 2020 - Present. Michael serves as the statistical lead for USEPA's National Aquatic Resource Surveys (NARS). The NARS, in collaboration with all 50 US states as well as tribes and other partners, monitor the chemical and biological integrity of US waters, including lakes/reservoirs, rivers/streams, wetlands, and coastal waters.

Michael Dumelle received his Ph.D. in statistics from Oregon State University in 2020. He is currently an author of five CRAN `R` packages for spatial data analysis including `spsurvey`, `spmodel`, `SSN2`, `SSNbler`, and `sptotal`. He research interests are in spatial statistics, survey design, software development, and statistical applications to ecological data.

Appendix S2 Relationship between $\mathbf{Y}$, $\mathbf{y}$, $\mathbf{R}$, and $\mathbf{r}$

The matrices $\mathbf{Y}$, $\mathbf{R}$, $\mathbf{Y}_{obs}$, and $\mathbf{Y}_{mis}$ are very general and represent all possible observed items as well as missingness patterns. In practice, however, we only see one data set with a single set of observed items and a single missingness pattern. We indicate the values $\mathbf{Y}$, $\mathbf{R}$, $\mathbf{Y}_{obs}$, and $\mathbf{Y}_{mis}$ which are realized (i.e., seen) for a single data set as $\mathbf{y}$, $\mathbf{r}$, $\mathbf{y}_{obs}$, and $\mathbf{y}_{mis}$, respectively. To make these descriptions more concrete, we provide an additional example here that relies on familiarity with the Gaussian (i.e., normal) and Poisson distributions of random variables and some elementary `R` code for simulating these variables.

Suppose an analyst is studying two variables $\mathbf{Y}_1$ and $\mathbf{Y}_2$ for each of six units. Further suppose that each element of $\mathbf{Y}_1$ is an independent and identically distributed (i.e., i.i.d.) Gaussian random variable with a mean of zero and a variance of one, and that each element of $\mathbf{Y}_2$ is an i.i.d. Poisson random variable with a mean (i.e., rate) of two. The matrix $\mathbf{Y}$ has six rows (one for each unit) and two columns (one for each variable, $\mathbf{Y}_1$ and $\mathbf{Y}_2$), totaling 12 items. The matrix $\mathbf{Y}$ represents all possible realizations of these 12 items. A single realization of both $\mathbf{Y}_1$ and $\mathbf{Y}_2$ yield $\mathbf{y}_1$ and $\mathbf{y}_2$, respectively, which together compose $\mathbf{y}$. The relationship between $\mathbf{Y}$ and $\mathbf{y}$ is made more concrete via the following `R` example. We first create a function called `bold_y` that simulate a single realization of $\mathbf{y}_1$ and $\mathbf{y}_2$ and places them into the $\mathbf{y}$ matrix:

```
bold_y <- function() {
  y_1 <- rnorm(n = 6, mean = 0, sd = 1)
  y_2 <- rpois(n = 6, lambda = 2)
  y <- matrix(c(y_1, y_2), nrow = 6, ncol = 2)
  return(y)
}
```

A single realization of $\mathbf{y}$ looks like:

```
set.seed(1)
y_draw_one <- bold_y()
```

```
print(y_draw_one)
#>             [,1] [,2]
#> [1,] -0.6264538    3
#> [2,]  0.1836433    1
#> [3,] -0.8356286    3
#> [4,]  1.5952808    2
#> [5,]  0.3295078    3
#> [6,] -0.8204684    6
```

Another realization of $\mathbf{y}$ looks like:

```
y_draw_two <- bold_y()
print(y_draw_two)
#>             [,1] [,2]
#> [1,] -0.3053884    2
#> [2,]  1.5117812    2
#> [3,]  0.3898432    2
#> [4,] -0.6212406    1
#> [5,] -2.2146999    3
#> [6,]  1.1249309    2
```

We can approximate the distribution of $\mathbf{Y}$ using the `approx_bold_Y` function, which simulates `s` different $\mathbf{y}$'s

```
approx_bold_Y <- function(s) {
  Y <- replicate(s, bold_y())
  return(Y)
}
```

The larger the `s`, the closer the approximation to $\mathbf{Y}$.

The following code generates 10,000 realizations of $\mathbf{y}$ and places them into an an array. The dimensions of the array are $6 \times 2 \times 10,000$:

```
set.seed(1)
approx_bold_Y_draw <- approx_bold_Y(s = 10000)
```

The first realization (6×2 array element) of $\mathbf{y}$ matches `y_draw_one`:

```
print(approx_bold_Y_draw[, , 1])
#>             [,1] [,2]
#> [1,] -0.6264538    3
#> [2,]  0.1836433    1
#> [3,] -0.8356286    3
#> [4,]  1.5952808    2
#> [5,]  0.3295078    3
#> [6,] -0.8204684    6
```

The 10,000th realization of of $\mathbf{y}$ is

```
print(approx_bold_Y_draw[, , 10000])
#>             [,1] [,2]
#> [1,]  1.2816568    3
#> [2,] -0.8675607    1
#> [3,]  0.4449571    2
#> [4,]  1.3997355    4
#> [5,] -0.2384143    2
#> [6,]  0.1294678    4
```

The first 40 realizations of $\mathrm{y}_{1,1}$ are

```
head(approx_bold_Y_draw[1, 1, ], n = 40)
#>  [1] -0.62645381 -0.30538839  0.82122120 -1.47075238 -0.39428995 -0.70749516
```

```
#>  [7]  1.43302370  0.02800216  0.61072635 -0.13517862 -0.54252003 -0.47340064
#> [13]  0.38418536 -0.27911330 -0.07356440 -1.53644982 -1.11592011 -0.92936215
#> [19]  1.05848305  0.10580237 -1.23132342 -0.92610950  0.41197471  0.54132734
#> [25] -0.30874057  1.80314191 -0.05972328 -0.87870761 -0.06965481 -0.38923718
#> [31] -0.25893258  0.91980368  0.43370215  0.95513668  0.08296573  1.06916146
#> [37]  0.48202950  1.36443493  0.24621084  0.67726849
```

A histogram of all 10,000 realizations of $y_{1,1}$ may be obtained from

```
# output omitted
hist(approx_bold_Y_draw[1, 1, ])
```

The matrices $\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}$, $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}$ are similarly structured except that they represent i.i.d. Bernoulli random variables with outcomes of one (observed) or zero (missing). Let $\mathbf{R}_1$ be an i.i.d. Bernoulli random variable with probability of observance 0.9 and $\mathbf{R}_2$ be an i.i.d. Bernoulli random variable with probability of observance 0.7. We first create a function called `bold_r` that simulates a single realization of $\mathbf{r}_1$ and $\mathbf{r}_2$ and places them into the $\mathbf{r}$ matrix.

```
bold_r <- function() {
    r_1 <- rbinom(n = 6, size = 1, prob = 0.9)
    r_2 <- rbinom(n = 6, size = 1, prob = 0.7)
    r <- matrix(c(r_1, r_2), nrow = 6, ncol = 2)
    return(r)
}
```

A single realization of $\mathbf{r}$ looks like:

```
set.seed(1)
r_draw_one <- bold_r()
print(r_draw_one)
#>      [,1] [,2]
```

```
#> [1,]    1    0
#> [2,]    1    1
#> [3,]    1    1
#> [4,]    0    1
#> [5,]    1    1
#> [6,]    1    1
```

Another realization of $\mathbf{r}$ looks like:

```
r_draw_two <- bold_r()
print(r_draw_two)
#>      [,1] [,2]
#> [1,]    1    1
#> [2,]    1    0
#> [3,]    1    0
#> [4,]    1    1
#> [5,]    1    1
#> [6,]    0    1
```

We can approximate the distribution of $\mathbf{R}$ using the `approx_bold_R` function, which simulates `s` different $\mathbf{r}$'s

```
approx_bold_R <- function(s) {
    R <- replicate(s, bold_r())
    return(R)
}
```

The larger the `s`, the closer the approximation to $\mathbf{R}$.

The following code generates 10,000 realizations of $\mathbf{r}$ and places them into an an array. The dimensions of the array are $6 \times 2 \times 10,000$:

```
set.seed(1)
approx_bold_R_draw <- approx_bold_R(s = 10000)
```

The first realization (6×2 array element) of $\mathbf{r}$ matches `r_draw_one`:

```
print(approx_bold_R_draw[, , 1])
#>      [,1] [,2]
#> [1,]    1    0
#> [2,]    1    1
#> [3,]    1    1
#> [4,]    0    1
#> [5,]    1    1
#> [6,]    1    1
```

The 10,000th realization of of $\mathbf{r}$ is

```
print(approx_bold_R_draw[, , 10000])
#>      [,1] [,2]
#> [1,]    1    1
#> [2,]    1    0
#> [3,]    1    1
#> [4,]    1    1
#> [5,]    1    1
#> [6,]    1    0
```

The first 40 realizations of $\mathrm{r}_{1,1}$ are

```
head(approx_bold_R_draw[1, 1, ], n = 40)
#>  [1] 1 1 1 1 1 0 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
#> [39] 1 1
```

A histogram of all 10,000 realizations of $\mathrm{r}_{1,1}$ may be obtained from

```
# output omitted
hist(approx_bold_R_draw[1, 1, ])
```

Subsetting each $\mathbf{y}$ by its corresponding $\mathbf{r}$ yields $\mathbf{y}_{obs}$ (for $\mathbf{r}$ elements that equal one) and $\mathbf{y}_{mis}$ (for $\mathbf{r}$ elements that equal zero).

Appendix S3 Details Regarding the Simulation Study

S3.1 An Inferential Comparison of Various Missing Data Methods

Here we provide details for Type-1 predictive mean matching and the Fully Bayesian Data Augmentation model specification from the simulation study in the *An Inferential Comparison of Various Missing Data Methods* section of the manuscript.

S3.1.1 Type-1 Predictive Mean Matching

Predictive mean matching chooses a donor for imputation by comparing the closeness between the fitted value of an item for each unit in the complete data and the fitted value for an item requiring imputation. The fitted value for the ith item from the complete data is $\hat{\mathrm{y}}_i$ and the fitted value for the item from the new unit is $\hat{\mathrm{y}}_{imp}$. Type-1 predictive mean matching uses a linear regression formulation to determine $\hat{\mathrm{y}}_i$ and $\hat{\mathrm{y}}_{imp}$:

$$\hat{\mathrm{y}}_i = \mathbf{x}_i\hat{\boldsymbol{\beta}} \text{ and } \hat{\mathrm{y}}_{imp} = \mathbf{x}_{imp}\tilde{\boldsymbol{\beta}}$$

where $\hat{\boldsymbol{\beta}}$ is an estimate of $\boldsymbol{\beta}$ (e.g., from ordinary least squares, posterior median, etc.) and $\tilde{\boldsymbol{\beta}}$ is a draw from the posterior distribution of $\boldsymbol{\beta}$ based on a Bayesian linear regression model (White et al., 2011; Van Buuren, 2018).

S3.1.2 Fully Bayesian Data Augmentation Model Specification

Let $\mathrm{r}_i^{\mathrm{y}}$ be the observation indicator for y_i and let $\mathrm{r}_i^{\mathrm{x}_2}$ be the observation indicator for $\mathrm{x}_{2,i}$. Let $(1, \mathrm{x}_{1,i}, \mathrm{x}_{2,i})$ be the row-vector of predictors for unit i. Let $[\mathrm{y}_i \mid \mu_i, \sigma^2]$ be the (univariate) Gaussian density function corresponding with mean μ_i and variance σ^2 evaluated at y. The Fully Bayesian Data Augmentation (FBDA) model specification used in the simulation study treats missing y_i and $\mathrm{x}_{2,i}$ as unknown parameters; thus we denote these parameters as $\dot{\mathrm{y}}_i$ and $\dot{\mathrm{x}}_{2,i}$. We assume the

following generative model for our data:

$$
\begin{aligned}
[\mathrm{x}_{2,i} \mid \mu_x, \sigma_x^2] &\sim \text{Gaussian}(\mu_x, \sigma_x^2) \\
[\mathrm{y}_i \mid \mathrm{x}_{1,i}, \mathrm{x}_{2,i}, \boldsymbol{\beta}, \sigma_y^2] &\sim \text{Gaussian}((1, \mathrm{x}_{1,i}, \mathrm{x}_{2,i})\boldsymbol{\beta}, \sigma_y^2)
\end{aligned}
\tag{1}
$$

The full set of parameters is as follows:

$$
\boldsymbol{\beta}, \sigma_y^2, \sigma_x^2, \mu_x, \{\dot{\mathrm{y}}_i \mid \mathrm{r}_i^{\mathrm{y}} = 0, i = 1, \ldots, n\}, \{\dot{\mathrm{x}}_{2,i} \mid \mathrm{r}_i^{\mathrm{x}} = 0, i = 1, \ldots, n\}.
$$

We subsequently use the abbreviations $\{\dot{\mathrm{y}}_i\} \equiv \{\dot{\mathrm{y}}_i \mid \mathrm{r}_i^{\mathrm{y}} = 0, i = 1, \ldots, n\}$ and $\{\dot{\mathrm{x}}_{2,i}\} \equiv \{\dot{\mathrm{x}}_{2,i} \mid \mathrm{r}_i^{\mathrm{x}} = 0, i = 1, \ldots, n\}$. Under the assumed model (1), the likelihood for the unknown parameters $\boldsymbol{\beta}, \sigma_y^2, \sigma_x^2, \mu_x, \{\dot{\mathrm{y}}_i\}, \{\dot{\mathrm{x}}_{2,i}\}$ is as follows:

$$
\begin{aligned}
&\prod_{i \mid \mathrm{r}_i^{\mathrm{y}}=1, \mathrm{r}_i^{X}=1} [\mathrm{y}_i \mid (1, \mathrm{x}_{1,i}, \mathrm{x}_{2,i})\boldsymbol{\beta}, \sigma_y^2][\mathrm{x}_{2,i} \mid \mu_x, \sigma_x^2] \\
\times &\prod_{i \mid \mathrm{r}_i^{\mathrm{y}}=1, \mathrm{r}_i^{\mathrm{x}}=0} [\mathrm{y}_i \mid (1, \mathrm{x}_{1,i}, \dot{\mathrm{x}}_{2,i})\boldsymbol{\beta}, \sigma_y^2] \\
\times &\prod_{i \mid \mathrm{r}_i^{\mathrm{y}}=0, \mathrm{r}_i^{\mathrm{x}}=1} [\mathrm{x}_{2,i} \mid \mu_x, \sigma_x^2]
\end{aligned}
$$

This likelihood coupled with the priors implied for unknown parameters $\{\dot{\mathrm{y}}_i\}, \{\dot{\mathrm{x}}_{2,i}\}$ and the unknown parameters $\boldsymbol{\beta}, \sigma_y^2, \mu_x, \sigma_x^2$ yields the joint posterior over the unknown parameters:

$$
\begin{aligned}
p[\boldsymbol{\beta}, \sigma_y^2, \sigma_x^2, \mu_x, \{\dot{\mathrm{y}}_i\}, \{\dot{x}_{2,i}\} \mid \mathrm{y}_{(1)}, \mathrm{x}_1, \mathrm{x}_{2(1)}] \propto & \prod_{i|\mathrm{r}_i^{\mathrm{y}}=1,\mathrm{r}_i^{\mathrm{x}}=1} [\mathrm{y}_i \mid (1, \mathrm{x}_{1,i}, \mathrm{x}_{2,i})\boldsymbol{\beta}, \sigma_y^2][\mathrm{x}_{2,i} \mid \mu_x, \sigma_x^2] \\
& \times \prod_{i|\mathrm{r}_i^{\mathrm{y}}=1,\mathrm{r}_i^{\mathrm{x}}=0} [\mathrm{y}_i \mid (1, \mathrm{x}_{1,i}, \dot{\mathrm{x}}_{2,i})\boldsymbol{\beta}, \sigma_y^2][\dot{\mathrm{x}}_{2,i} \mid \mu_x, \sigma_x^2] \\
& \times \prod_{i|\mathrm{r}_i^{\mathrm{y}}=0,\mathrm{r}_i^{\mathrm{x}}=1} [\dot{\mathrm{y}}_i \mid (1, \mathrm{x}_{1,i}, \mathrm{x}_{2,i})\boldsymbol{\beta}, \sigma_y^2][\mathrm{x}_{2,i} \mid \mu_x, \sigma_x^2] \\
& \times \prod_{i|\mathrm{r}_i^{\mathrm{y}}=0,\mathrm{r}_i^{\mathrm{x}}=0} [\dot{\mathrm{y}}_i \mid (1, \mathrm{x}_{1,i}, \dot{\mathrm{x}}_{2,i})\boldsymbol{\beta}, \sigma_y^2][\dot{\mathrm{x}}_{2,i} \mid \mu_x, \sigma_x^2] \\
& \times \pi[\boldsymbol{\beta}]\pi[\sigma_y][\sigma_x][\mu_x].
\end{aligned}
$$

The priors $\pi[\boldsymbol{\beta}]$, $\pi[\mu_x]$, $\pi[\sigma_y]$, and $\pi[\sigma_x]$ are `brms`'s default priors for the slope parmaeters parameters, means, and standard deviations, respectively (Bürkner, 2017).

S3.2 Revisiting the Inferential Comparison

The 13 missing data methods used in the simulation study are relisted in Table S1.

S3.2.1 Imputation Performance

In each simulation trial, we computed statistics like mean bias (Mbias) and root-mean-squared error (RMSE) of the imputations themselves for the $\mathbf{x}_2$ variable. More formally, imputation Mbias is

$$
\text{MBias}(\text{Imp}(\mathbf{x}_2)) = \frac{1}{2000 \times J} \sum_{i=1}^{2{,}000} \sum_j (\tilde{\mathrm{x}}_{2,j} - \mathrm{x}_{2,j}),
$$

where i indexes the simulation trials, j indexes the J missing items for $\mathbf{x}_2$ within a simulation trial (which vary from trial to trial), $\tilde{\mathrm{x}}_{2,j}$ is the imputed value of x_2 and $\mathrm{x}_{2,j}$ is the true value of x_2. Similarly, imputation RMSPE is

$$
\text{RMSE}(\text{Imp}(\mathbf{x}_2)) = \sqrt{\frac{1}{2000 \times J} \sum_{i=1}^{2{,}000} \sum_j (\tilde{\mathrm{x}}_{2,j} - \mathrm{x}_{2,j})^2}.
$$

Table S2 shows imputation performance for $\mathbf{x}_2$ in all 13 missing data methods. Recall that the methods best at recreating the imputations (e.g., Mean, Reg) tended to perform the worst at estimating the regression slope parameters β_1 and β_2.

S3.3 A Predictive Comparison of Various Missing Data Methods

Here we derive prediction standard errors in the (nonspatial) linear model and review imputation performance for $\mathbf{y}$.

S3.3.1 Deriving Prediction Standard Errors in the Linear Model

The linear regression model with independent and identically distributed (i.i.d) errors can be expressed in matrix notation as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

where $\mathbf{y}$ is an $n \times 1$ response vector, $\mathbf{X}$ is an $n \times p$ matrix of explanatory variables, $\boldsymbol{\beta}$ is a $p \times 1$ vector of slope parameters (i.e., fixed effects) that control the average impact of each explanatory variable (i.e. column of $\mathbf{X}$) on $\mathbf{y}$, and $\boldsymbol{\epsilon}$ is a vector of i.i.d. random errors with mean zero and common variance σ^2. It can be shown that the best linear unbiased estimator of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

where $(\cdot)^T$ is the transpose operator and $(\cdot)^{-1}$ is the inverse operator. It can be shown that the variance of $\hat{\boldsymbol{\beta}}$ is $\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$ (Rencher and Schaalje, 2008).

Suppose that $\mathbf{x}_{new}$ is a $1 \times p$ vector of explanatory variable values for a new unit and the goal is to use $\mathbf{x}_{new}$ to make a corresponding prediction of y_{new}. The best linear unbiased predictor of y_{new} is $\mathbf{x}_{new}\hat{\boldsymbol{\beta}}$ with corresponding prediction variance given by (Rencher and Schaalje, 2008)

$$\mathrm{Var}(\mathrm{y}_{new} - \mathbf{x}_{new}\hat{\boldsymbol{\beta}}) = \sigma^2 + \sigma^2\mathbf{x}_{new}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_{new}^T.$$

When σ^2 is unknown, σ^2 above is replace by an estimate, $\hat{\sigma}^2$.

S3.3.2 Imputation Performance

In each simulation trial, we computed statistics like mean bias (Mbias) and root-mean-squared error (RMSE) of the imputations themselves for the $\mathbf{y}$ variable. More formally, imputation Mbias is

$$\text{MBias}(\text{Imp}(\mathbf{y})) = \frac{1}{2000 \times J} \sum_{i=1}^{2{,}000} \sum_{j} (\tilde{\text{y}}_j - \text{y}_j),$$

where i indexes the simulation trials, j indexes the J missing items for $\mathbf{y}_2$ within a simulation trial (which vary from trial to trial), $\tilde{\text{y}}_{2,j}$ is the imputed value of y_2 and $\text{y}_{2,j}$ is the true value of y_2. Similarly, imputation RMSPE is

$$\text{RMSE}(\text{Imp}(\mathbf{y})) = \sqrt{\frac{1}{2000 \times J} \sum_{i=1}^{2{,}000} \sum_{j} (\tilde{\text{y}}_j - \text{y}_j)^2}.$$

Table S2 shows imputation performance for $\mathbf{y}$ in all 13 missing data methods. Recall that the methods best at recreating the imputations (e.g., Mean, Reg) tended to perform the worst at predicting y_{new}.

References

Bürkner, P.C., 2017. brms: An R package for bayesian multilevel models using stan. Journal of Statistical Software 80, 1–28.

Rencher, A.C., Schaalje, G.B., 2008. Linear models in statistics. John Wiley & Sons.

Van Buuren, S., 2018. Flexible imputation of missing data. CRC press.

White, I.R., Royston, P., Wood, A.M., 2011. Multiple imputation using chained equations: Issues and guidance for practice. Statistics in Medicine 30, 377–399.

Tables

Method	Abbreviation	Imputations	Parameters
Complete Case Analysis	CCA	-	-
Mean Imputation	Mean	Deterministic	Deterministic
Regression Imputation	Reg	Deterministic	Deterministic
Nearest Neighbor Imputation	NN	Deterministic	Deterministic
Stochastic Regression Imputation (Single)	StReg-S	Random	Deterministic
Predictive Mean Matching Type-0 Imputation (Single)	PMMT0-S	Random	Deterministic
Bootstrap Regression Imputation (Single)	Boot-S	Random	Random
Predictive Mean Matching Type-1 Imputation (Single)	PMMT1-S	Random	Random
Random Regression Imputation (Multiple)	StReg-M	Random	Deterministic
Predictive Mean Matching Type-0 Imputation (Multiple)	PMMT0-M	Random	Deterministic
Bootstrap Regression Imputation (Multiple)	Boot-M	Random	Random
Predictive Mean Matching Type-1 Imputation (Multiple)	PMMT1-M	Random	Random
Fully Bayesian Data Augmentation	FBDA	Random	Random

Table S1: Missing data methods and abbreviations used in the simulation study. The "Imputations" column clarifies whether the imputation process is deterministic (i.e., nonrandom) or has a random component. The "Parameters" column clarifies whether the parameters used to fill in missing data items are deterministic or have a random component.

	Imp($\mathbf{x}_2$)	
Approach	MBias	RMSE
CCA	–	–
Mean	-0.00	1.01
Reg	-0.00	0.92
NN	-0.02	1.31
StReg-S	-0.01	1.26
PMMT0-S	0.00	1.23
Boot-S	-0.00	1.25
PMMT1-S	-0.03	1.25
StReg-M	-0.00	1.25
PMMT0-M	-0.00	1.23
Boot-M	0.00	1.25
PMMT1-M	-0.03	1.24
FBDA	0.00	1.26

Table S2: Performance across 2,000 independent simulation trials for imputations of $\mathbf{x}_2$ themselves (Imp($\mathbf{x}_2$)). Mean bias (MBias) and root-mean-squared error (RMSE) are reported.

	Imp($\mathbf{y}$)	
Method	MBias	RMSE
CCA	–	–
Mean	-0.35	1.54
Reg	0.00	1.30
NN	-0.02	1.85
StReg-S	0.01	1.78
PMMT0-S	-0.02	1.74
Boot-S	0.00	1.77
PMMT1-S	-0.03	1.76
StReg-M	0.00	1.77
PMMT0-M	-0.01	1.74
Boot-M	0.01	1.77
PMMT1-M	-0.02	1.76
FBDA	-0.00	1.82

Table S3: Performance across 2,000 independent simulation trials for imputations of $\mathbf{y}$ themselves (Imp($\mathbf{y}$)). Mean bias (MBias) and root-mean-squared error (RMSE) are reported.