Adv. Stat. Topics A - Missing data

Afternoon session

Anne Helby Petersen

Program outline

- 12.00-12.50: Imputation and Multiple Imputation using Chained Equations (MICE)
- 12.50-14.15: Work with data: Data analysis with missing information
- 14.15-14.45: Presentations
- 14.45-15.00: Further perspectives and more resources

Imputation

- Imputation: Fill in missing slots in the data with plausible values.

Imputation

- Imputation: Fill in missing slots in the data with plausible values.
- Terrible idea

Imputation

- Imputation: Fill in missing slots in the data with plausible values.
- Terrible idea... if you do it just once.

Imputation

- Imputation: Fill in missing slots in the data with plausible values.
- Terrible idea... if you do it just once.
- Wonderful idea if you do it multiple times.

Example: Simple missing information setup

- Imagine that we wish to estimate the effect of X on Y, controlling for Z.
- X suffers from missing information (MCAR). Assume that we order the observations such that $X_1, ..., X_d$ have missing information, while $X_{d+1}, ..., X_n$ are fully observed.
- Assume that Y and Z are all fully observed.

Example: Simple missing information setup

- Imagine that we wish to estimate the effect of X on Y, controlling for Z.
- X suffers from missing information (MCAR). Assume that we order the observations such that $X_1, ..., X_d$ have missing information, while $X_{d+1}, ..., X_n$ are fully observed.
- Assume that Y and Z are all fully observed.
- Note: Complete case analysis would produce an unbiased, but inefficient estimate.

Simulating a small dataset in R

Simulating a small dataset in R

```r
n <- 200
set.seed(1331)
Z <- rnorm(n, sd = 1)
X <- Z + rnorm(n, sd = 1)
Y <- 2*X + Z + rnorm(n, sd = 2)
```

Simulating a small dataset in R

```r
n <- 200
set.seed(1331)
Z <- rnorm(n, sd = 1)
X <- Z + rnorm(n, sd = 1)
Y <- 2*X + Z + rnorm(n, sd = 2)
```

```r
true_X <- X
true_xmean <- mean(X)
true_xsd <- sd(X)
true_model <- lm(Y ~ X + Z)
```

Simulating a small dataset in R

```
n <- 200
set.seed(1331)
Z <- rnorm(n, sd = 1)
X <- Z + rnorm(n, sd = 1)
Y <- 2*X + Z + rnorm(n, sd = 2)
```

```
true_X <- X
true_xmean <- mean(X)
true_xsd <- sd(X)
true_model <- lm(Y ~ X + Z)
```

```
d <- 40
X[1:d] <- NA
```

Simulating a small dataset in R

```r
n <- 200
set.seed(1331)
Z <- rnorm(n, sd = 1)
X <- Z + rnorm(n, sd = 1)
Y <- 2*X + Z + rnorm(n, sd = 2)
```

```r
true_X <- X
true_xmean <- mean(X)
true_xsd <- sd(X)
true_model <- lm(Y ~ X + Z)
```

```r
d <- 40
X[1:d] <- NA
```

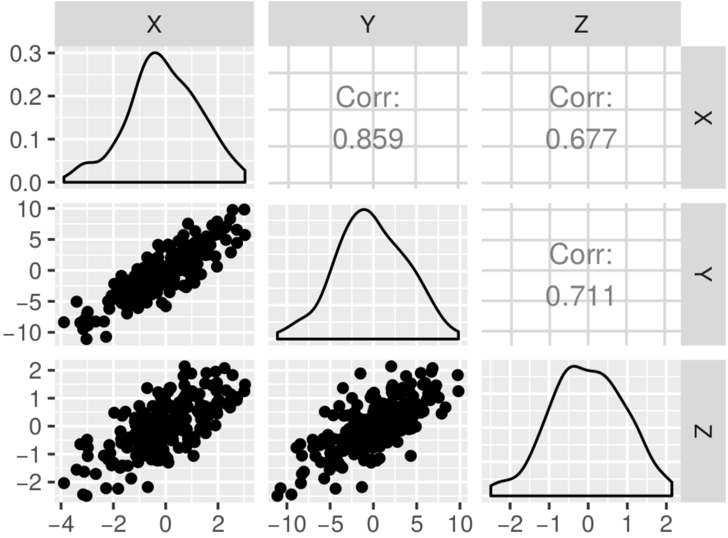
```r
X[36:40]
```

```
## [1] NA NA NA NA NA
```

```r
X[41:45]
```

```
## [1] -0.9404489  0.7807026  1.9016603 -0.3728711 -0.5331431
```

A quick overview of the data (no missing info.)



Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, ..., X_n$ into all $X_1, ..., X_d$.

Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, ..., X_n$ into all $X_1, ..., X_d$.

```
X_meanimp <- X
```

Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, ..., X_n$ into all $X_1, ..., X_d$.

```
X_meanimp <- X
```

```
xobs_mean <- mean(X[(d+1):n])
```

Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, ..., X_n$ into all $X_1, ..., X_d$.

```
X_meanimp <- X
```

```
xobs_mean <- mean(X[(d+1):n])
```

```
X_meanimp[1:d] <- xobs_mean
```

Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, \ldots, X_n$ into all $X_1, \ldots, X_d$.

```
X_meanimp <- X
```

```
xobs_mean <- mean(X[(d+1):n])
```

```
X_meanimp[1:d] <- xobs_mean
```

```
#Compare mean for full X with mean of mean imputed X
true_xmean; mean(X_meanimp)
```

```
## [1] -0.07813252
```

```
## [1] -0.1551874
```

Mean imputation (1/3)

Mean imputation: Insert the mean (or mode) of $X_{\mathrm{d}+1}, \ldots, X_n$ into all $X_1, \ldots, X_d$.

```
X_meanimp <- X
```

```
xobs_mean <- mean(X[(d+1):n])
```

```
X_meanimp[1:d] <- xobs_mean
```

```
#Compare mean for full X with mean of mean imputed X
true_xmean; mean(X_meanimp)
```

```
## [1] -0.07813252
```

```
## [1] -0.1551874
```

```
#Compare sd for full X with sd of mean imputed X
true_xsd; sd(X_meanimp)
```

```
## [1] 1.368721
```

```
## [1] 1.240263
```

Mean imputation (2/3)

Comparing model coefficients:

Mean imputation (2/3)

Comparing model coefficients:

```
round(summary(true_model)$coefficients,4)
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

Mean imputation (2/3)

Comparing model coefficients:

```r
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

```r
round(summary(lm(Y ~ X_meanimp + Z))$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0709     0.1623  0.4371   0.6626
## X_meanimp     1.7887     0.1639 10.9102   0.0000
## Z             1.6266     0.2150  7.5675   0.0000
```

Mean imputation (2/3)

Comparing model coefficients:

```
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

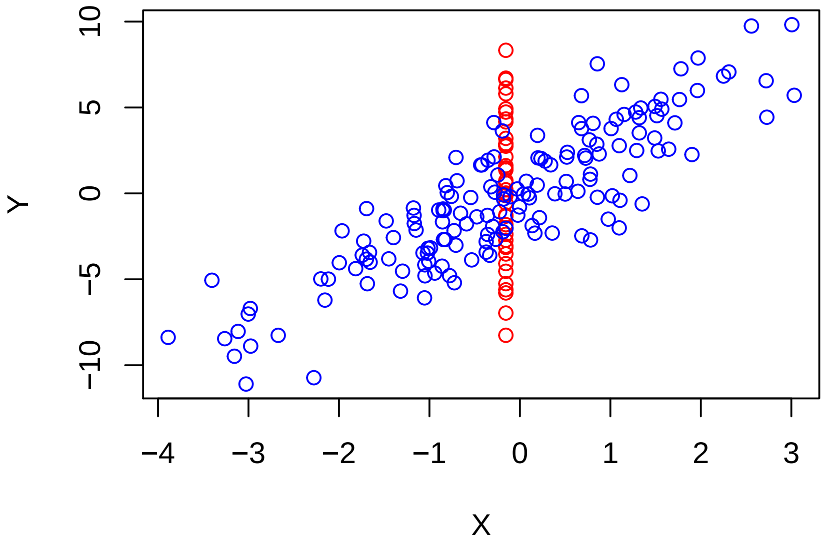
```
round(summary(lm(Y ~ X_meanimp + Z))$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0709     0.1623  0.4371   0.6626
## X_meanimp     1.7887     0.1639 10.9102   0.0000
## Z             1.6266     0.2150  7.5675   0.0000
```

Conclusion: Don't do mean imputation!

Mean imputation (3/3)

```
plot(Y ~ X_meanimp, xlab = "X",
     col = c(rep("red", 40), rep("blue", 160)))
```



Hot deck imputation (1/3)

Hot deck imputation (simplest version): For each missing value, $X_1, ..., X_d$, pick and insert a random value among the observed values $X_{\mathrm{d}+1}, ..., X_n$.

Hot deck imputation (1/3)

Hot deck imputation (simplest version): For each missing value, $X_1, ..., X_d$, pick and insert a random value among the observed values $X_{\mathrm{d}+1}, ..., X_n$.

```
X_hdimp <- X
```

Hot deck imputation (1/3)

Hot deck imputation (simplest version): For each missing value, $X_1, \ldots, X_d$, pick and insert a random value among the observed values $X_{d+1}, \ldots, X_n$.

```
X_hdimp <- X
```

```
set.seed(13)
X_hdimp[1:d] <- sample(X[(d+1):n], size = d,
                       replace = TRUE)
```

Hot deck imputation (1/3)

Hot deck imputation (simplest version): For each missing value, $X_1, ..., X_d$, pick and insert a random value among the observed values $X_{\mathrm{d}+1}, ..., X_n$.

```
X_hdimp <- X
```

```
set.seed(13)
X_hdimp[1:d] <- sample(X[(d+1):n], size = d,
                        replace = TRUE)
```

```
#Compare mean for full X with mean of mean imputed X
true_xmean; mean(X_hdimp)
```

```
## [1] -0.07813252
```

```
## [1] -0.2030766
```

Hot deck imputation (1/3)

Hot deck imputation (simplest version): For each missing value, $X_1, \ldots, X_d$, pick and insert a random value among the observed values $X_{\mathrm{d}+1}, \ldots, X_n$.

```
X_hdimp <- X
```

```
set.seed(13)
X_hdimp[1:d] <- sample(X[(d+1):n], size = d,
                       replace = TRUE)
```

```
#Compare mean for full X with mean of mean imputed X
true_xmean; mean(X_hdimp)
```

```
## [1] -0.07813252
```

```
## [1] -0.2030766
```

```
#Compare sd for full X with sd of mean imputed X
true_xsd; sd(X_hdimp)
```

```
## [1] 1.368721
```

```
## [1] 1.387267
```

Hot deck imputation (2/3)

Comparing model coefficients:

Hot deck imputation (2/3)

Comparing model coefficients:

```
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

Hot deck imputation (2/3)

Comparing model coefficients:

```r
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

```r
round(summary(lm(Y ~ X_hdimp + Z))$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0484     0.1781  0.2720   0.7859
## X_hdimp       1.2207     0.1492  8.1828   0.0000
## Z             2.1195     0.2188  9.6879   0.0000
```

Hot deck imputation (2/3)

Comparing model coefficients:

```
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

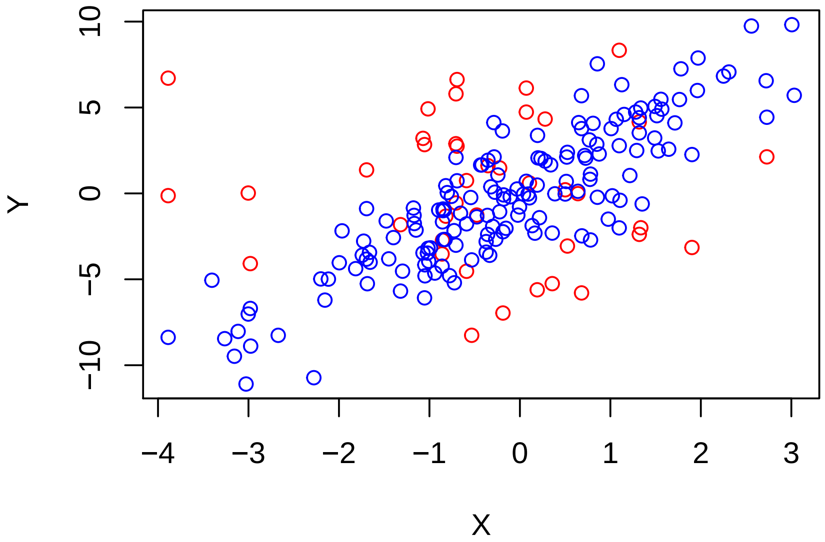
```
round(summary(lm(Y ~ X_hdimp + Z))$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0484     0.1781  0.2720   0.7859
## X_hdimp       1.2207     0.1492  8.1828   0.0000
## Z             2.1195     0.2188  9.6879   0.0000
```

Conclusion: Don't do hot deck imputation!

Hot deck imputation (3/3)

```r
plot(Y ~ X_hdimp, xlab = "X",
     col = c(rep("red", 40), rep("blue", 160)))
```



Regression imputation (1/4)

Regression imputation: Fit a regression model for all the observations, e.g.,

$$X_{\mathrm{i}} = \alpha + \beta_1 \cdot Y_i + \beta_2 \cdot Z_i + \epsilon_i$$

for $i = d + 1, ..., n$ and use this model to predict values for the remaining $X_1, ..., X_d$.

Regression imputation (1/4)

Regression imputation: Fit a regression model for all the observations, e.g.,

$$X_i = \alpha + \beta_1 \cdot Y_i + \beta_2 \cdot Z_i + \epsilon_i$$

for $i = d + 1, ..., n$ and use this model to predict values for the remaining $X_1, ..., X_d$.

```
obsdata <- data.frame(X = X[(d+1):n],
                      Y = Y[(d+1):n],
                      Z = Z[(d+1):n])
m_regimp <- lm(X ~ Y + Z, obsdata)
X_regimp <- X
X_regimp[1:d] <- predict(m_regimp,
                         newdata = data.frame(Y = Y[1:d],
                                              Z = Z[1:d]))
```

Regression imputation (2/4)

```
#Compare mean for full X with mean of reg. imputed X
true_xmean; mean(X_regimp)
```

```
## [1] -0.07813252
```

```
## [1] -0.1177343
```

```
#Compare sd for full X with mean of reg. imputed X
true_xsd; sd(X_regimp)
```

```
## [1] 1.368721
```

```
## [1] 1.352262
```

Regression imputation (3/4)

```r
round(summary(true_model)$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0532     0.1392 -0.3822   0.7028
## X             2.0756     0.1382 15.0158   0.0000
## Z             1.0260     0.2000  5.1297   0.0000
```

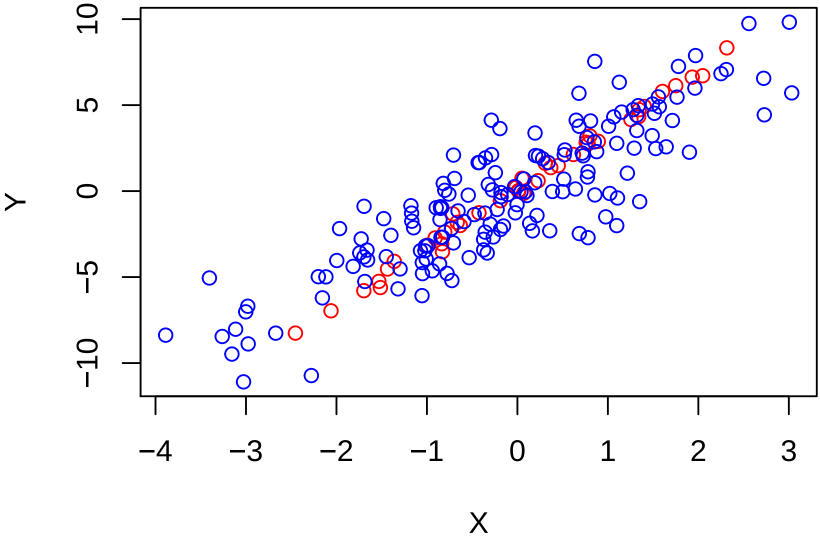
```r
round(summary(lm(Y ~ X_regimp + Z))$coefficients,4)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0505     0.1256  0.4024   0.6878
## X_regimp      2.2815     0.1264 18.0525   0.0000
## Z             0.8383     0.1807  4.6401   0.0000
```

Conclusion: Don't do regression imputation!

Regression imputation (4/4)

```
plot(Y ~ X_regimp, xlab = "X",
     col = c(rep("red", 40), rep("blue", 160)))
```



Stochastic regression imputation (1/2)

Stochastic regression imputation: Perform regression imputation, but add noise to the predictions by sampling from the residuals from the fitted model.

Stochastic regression imputation (1/2)

Stochastic regression imputation: Perform regression imputation, but add noise to the predictions by sampling from the residuals from the fitted model.

```
X_stocregimp <- X; set.seed(2)
X_stocregimp[1:d] <- X_regimp[1:d] +
  sample(residuals(m_regimp), size = d,
         replace = TRUE)
```

Stochastic regression imputation (1/2)

Stochastic regression imputation: Perform regression imputation, but add noise to the predictions by sampling from the residuals from the fitted model.

```
X_stocregimp <- X; set.seed(2)
X_stocregimp[1:d] <- X_regimp[1:d] +
  sample(residuals(m_regimp), size = d,
         replace = TRUE)
```

```
#Estimate from model with full X
round(summary(true_model)$coefficients,4)[2,]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##     2.0756     0.1382    15.0158     0.0000
```

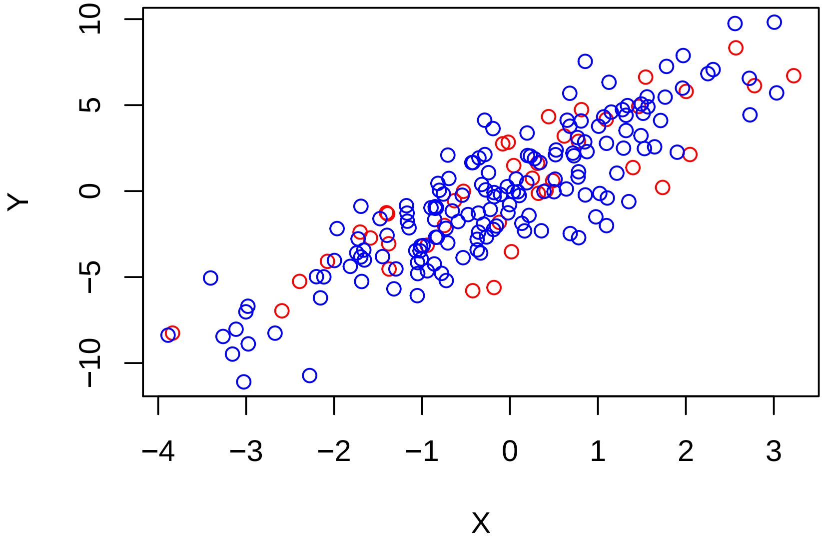
```
#Estimate from model with X imputed by stochastic regression
round(summary(lm(Y ~ X_stocregimp + Z))$coefficients,4)[2,]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##     2.0430     0.1309    15.6060     0.0000
```

Problem: The variance is still underestimated.

Stochastic regression imputation (2/2)

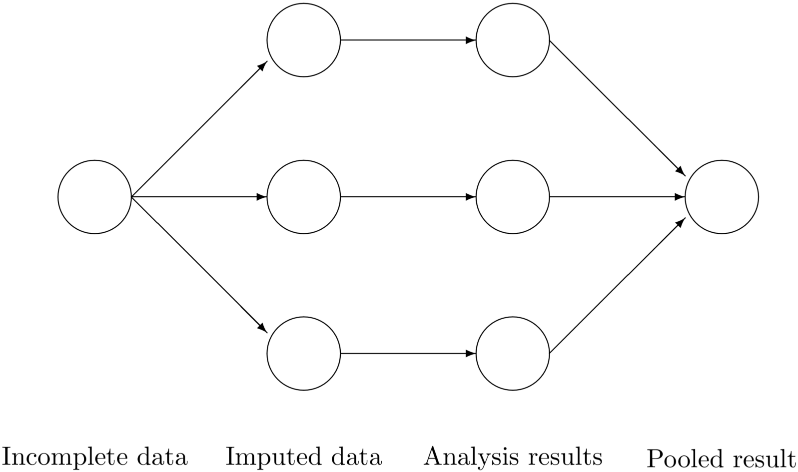
```
plot(Y ~ X_stocregimp, xlab = "X",
     col = c(rep("red", 40), rep("blue", 160)))
```



The problem with single imputation strategies

> Imputing one value for a missing datum cannot be correct in general, because we don't know what value to impute with certainty (if we did, it wouldn't be missing).
>
> — Donald B. Rubin

Multiple imputation



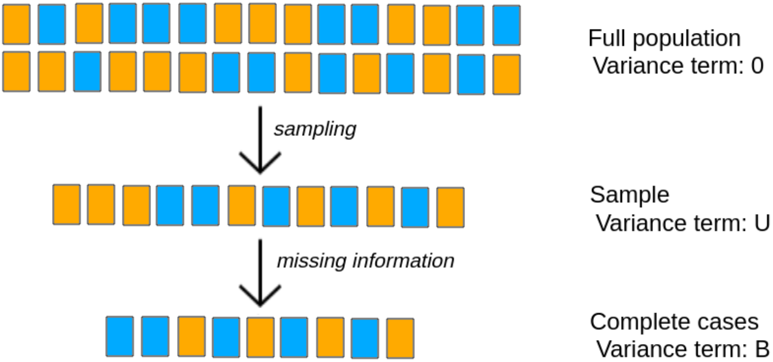
(Figure 1.6 from van Buuren 2019)

Variance under imputation

Recall: Variance measures the uncertainty of our estimate if we were to repeat the whole thing.

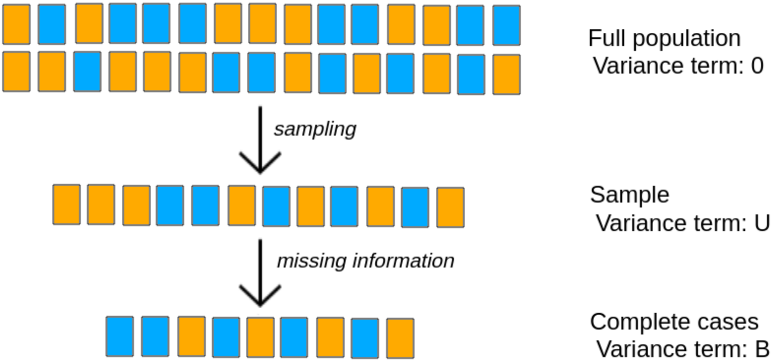
Variance under imputation

Recall: Variance measures the uncertainty of our estimate if we were to repeat the whole thing.



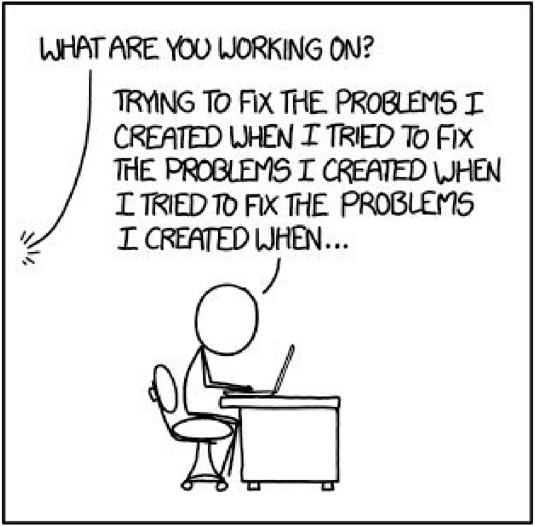
Variance under imputation

Recall: Variance measures the uncertainty of our estimate if we were to repeat the whole thing.



Problem: Variance accumulates; we need to use the (uncertain) sample to estimate the missing data model.

Variance accumulates



http://xkcd.com/1739/

Total variance (following van Buuren 2019)

It can be shown mathematically that

$$\text{Total variance} = U + B + B \cdot \frac{1}{m}$$

where m is the number of imputed datasets and

- U is the variance due to using a sample rather than the full population.
- B is the extra variance due to there being missing values.
- $B \cdot \frac{1}{m}$ is the extra variance due to having to estimate the missing data model.

The collective method for obtaining a correct estimate of the total variance (T) by use of multiple imputations is referred to as *Rubin's rules*.

Total variance (following van Buuren 2019)

It can be shown mathematically that

$$\text{Total variance} = U + B + B \cdot \frac{1}{m}$$

where m is the number of imputed datasets and

- U is the variance due to using a sample rather than the full population.
- B is the extra variance due to there being missing values.
- $B \cdot \frac{1}{m}$ is the extra variance due to having to estimate the missing data model.

The collective method for obtaining a correct estimate of the total variance (T) by use of multiple imputations is referred to as *Rubin's rules*.

Note: Larger m makes the last term small.

Multiple imputation by chained equations (MICE)

- A specific algorithm (method) for performing data analysis with missing information.
- Also known as imputation with *fully conditional specification* (FCS).
- Specifies imputation models variable-by-variable for each variable with missing information.
-

Multiple imputation by chained equations (MICE)

- A specific algorithm (method) for performing data analysis with missing information.
- Also known as imputation with *fully conditional specification* (FCS).
- Specifies imputation models variable-by-variable for each variable with missing information.
- Iteratively updates best guesses to allow all variables (even those with missing information) to inform the imputation of the others.

Multiple imputation by chained equations (MICE)

- A specific algorithm (method) for performing data analysis with missing information.
- Also known as imputation with *fully conditional specification* (FCS).
- Specifies imputation models variable-by-variable for each variable with missing information.
- **Iteratively updates best guesses to allow all variables (even those with missing information) to inform the imputation of the others.**

MICE: Sequential guessing (heuristically)

- Assume both X and Z have missing information.
- Let X_{obs} and Z_{obs} denote the observed values of X and Z, respectively.

MICE in R

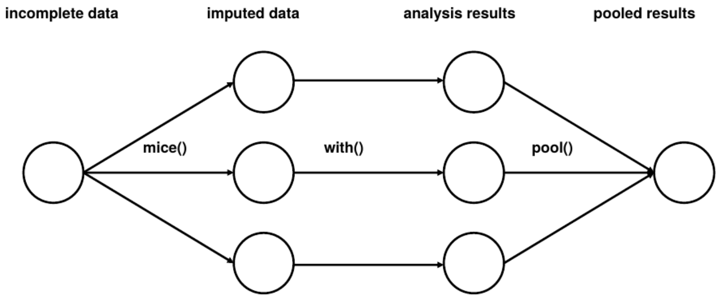
MICE is implemented in the `mice` package in R:

```
library(mice)
data <- data.frame(X = X, Y = Y, Z = Z)
set.seed(22)
imps <- mice(data, print = FALSE, m = 10)
fits <- with(imps, lm(Y ~ X + Z))
res <- pool(fits)

summary(res)[, c(1:3,6)]
```

```
##          term     estimate std.error       p.value
## 1 (Intercept) -0.007041764 0.1409787 0.9602335556908
## 2           X  2.072830228 0.1365646 0.0000000000000
## 3           Z  1.030463434 0.1992245 0.0000007903696
```

MICE in R: Schematic



(Figure 1 from van Buuren & Groothuis-Oudshoorn 2011)

MICE compared with stochastic regression imputation

```r
#Estimate from complete case analysis
round(summary(lm(Y ~ X + Z, data))$coefficients,4)[2,]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##     2.0358     0.1452    14.0252     0.0000
```

```r
#Estimate from model with X imputed by stochastic regression
round(summary(lm(Y ~ X_stocregimp + Z))$coefficients,4)[2,]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
##     2.0430     0.1309    15.6060     0.0000
```

```r
#Estimate from mice model (default settings)
round(summary(res)[2, c(2,3,4,6)],4)
```

```
##   estimate std.error statistic p.value
## 2   2.0728    0.1366   15.1784       0
```

Inspecting the variance components from mice

Note: `mice` delivers estimates of B (`b`), U (`ubar`), T (`t` $=$ `std.error`^{2}), as well as $\lambda = \frac{B\cdot(1+1/m)}{T}$ (`lambda`), riv $= \frac{B\cdot(1+1/m)}{U}$ (`riv`) and more:

```
> summary(res, type = "all")
         term  m     estimate std.error   statistic
1 (Intercept) 10 -0.007041764 0.1409787 -0.04994913
2           X 10  2.072830228 0.1365646 15.17839086
3           Z 10  1.030463434 0.1992245  5.17237362
        df      p.value       riv    lambda       fmi
1 141.1110 9.602336e-01 0.1228197 0.1093851 0.1217452
2 126.6437 0.000000e+00 0.1540145 0.1334598 0.1468278
3 138.9950 7.903696e-07 0.1271987 0.1128450 0.1253405
        ubar           b          t dfcom
1 0.01770097 0.001976389 0.01987499   197
2 0.01616087 0.002262735 0.01864988   197
3 0.03521153 0.004071692 0.03969039   197
```

Variable level imputation models

Default choices in `mice` package:

Numerical variables:

Predictive mean matching (`pmm`). A fusion between regression imputation and hot deck imputation: Use regression to find a selection of plausible “donor values”, choose one at random among them.

Categorical variables (>2 categories):

Multinomial logistic regression (`polyreg`). A regression imputation method.

Categorical variables ($=2$ categories):

Logistic regression (`logreg`). A regression imputation method.

Categorical variables (ordered categories):

Ordered logistic regression (`polr`). A regression imputation method.

Data exercise: Analyze `alcodata`

→ Go to "Exercise: Analyze" on the course website

https://biostatistics.dk/teaching/advtopicsA/notes.html

and work through the questions in small groups.

→ Add information to the Google slide show ("analyze") corresponding to your dataset - find the link in the exercises.

We will discuss your findings around 14:15.

Back to the Elderly study

Table 3.1: Estimated log odds ratios from the model of controlled consumption status using all full covariate adjustment. The reported estimates are on log odds ratio scale and they are computed relative to the following reference category: Treament MET; Gender male; Country Denmark; Age 60; Education none; No partner; Low ADS; Previous treatments 0. The mean log odds of having a controlled alcohol consumption in this reference group is represented by the intercept estimate. The reported p-values correspond to two-sided z-tests of the null-hypothesis of a zero parameter value.

	Estimate	Std. error	z statistic	p-value
Intercept	-0.3507	0.3050	-1.1499	0.2502
Treatment: MET+CRA	0.2028	0.1801	1.1260	0.2602
Country: USA	0.0736	0.2327	0.3164	0.7517
Country: Germany	-0.0351	0.2522	-0.1392	0.8893
Gender: Female	-0.5543	0.1906	-2.9085	0.0036
Age	0.0677	0.0211	3.2038	0.0014
Married or cohabiting: Yes	0.2270	0.1877	1.2094	0.2265
Severity: Intermediate	-0.0777	0.2307	-0.3367	0.7363
Severity: Substantial or severe	-0.2767	0.4096	-0.6755	0.4994
Education: At most undergraduate degree	0.0518	0.2286	0.2268	0.8206
Education: Graduate or post-graduate	-0.4463	0.2872	-1.5537	0.1202
Previous treatments: 1-2	0.2655	0.2187	1.2140	0.2247
Previous treatments: 3+	0.2938	0.3087	0.9517	0.3413

Elderly sensitivity analyses - models

We fitted five additional models:

MiD Missing is drinking approach: Treating all missing observations as relapsers (non-controlled consumption).

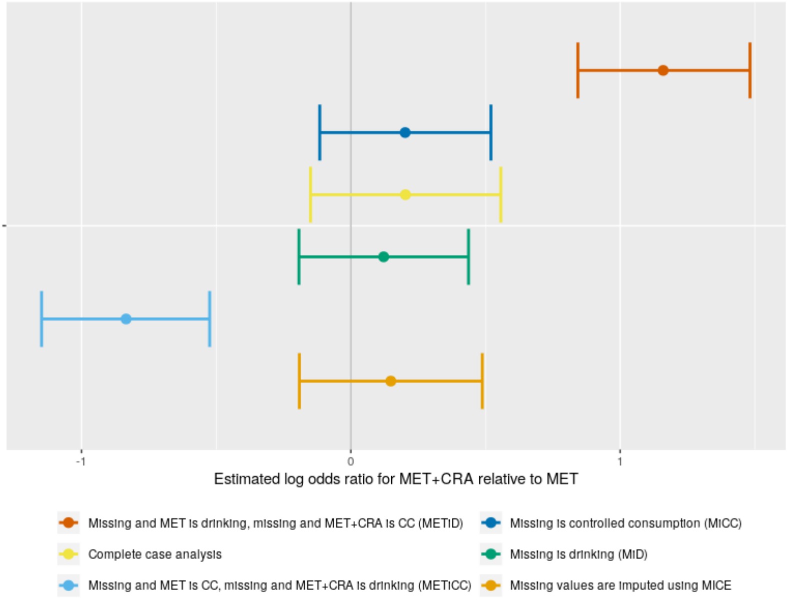
MiCC Missing is CC approach: Treating all missing observations as controlled consumption.

METiD MET is drinking approach: Treating missing observations for patients treated with MET as drinking, while missing obsevrations from MET+CRA-patients are treated as controlled consumption.

METiCC MET is CC: Treating missing observations for patients treated with MET+CRA as drinking, while missing observations from MET-patients are treated as controlled consumption.

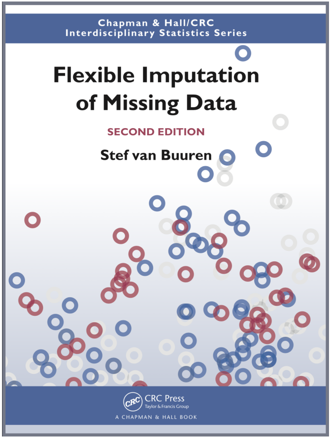
MICE Multiple imputation of missing observation using all variables from the primary model and controlled consumption information from previous time points.

Elderly sensitivity analyses - results



Further resources (1)

Excellent book by Stef van Buuren (2019)



https://stefvanbuuren.name/fimd/

Further resources (2)

Multiple imputation for Cox models:

STATISTICS IN MEDICINE
Statist. Med. 2009; **28**:1982–1998
Published online 19 May 2009 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/sim.3618

Imputing missing covariate values for the Cox model

Ian R. White[1,*,†] and Patrick Royston[2]

[1] *MRC Biostatistics Unit, Institute of Public Health, Robinson Way, Cambridge CB2 0SR, U.K.*
[2] *MRC Clinical Trials Unit, Cancer Group, London, U.K.*

SUMMARY

Multiple imputation is commonly used to impute missing data, and is typically more efficient than complete cases analysis in regression analysis when covariates have missing values. Imputation may be performed using a regression model for the incomplete covariates on other covariates and, importantly, on the outcome. With a survival outcome, it is a common practice to use the event indicator D and the log of the observed event or censoring time T in the imputation model, but the rationale is not clear.

https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.3618

Further resources (3)

Guideline for MICE in practice:

Tutorial in Biostatistics

Statistics in Medicine

Received 3 September 2009, Accepted 14 July 2010 Published online 30 November 2010 in Wiley Online Library

(wileyonlinelibrary.com) DOI: 10.1002/sim.4067

Multiple imputation using chained equations: Issues and guidance for practice

Ian R. White,[a*†] Patrick Royston[b] and Angela M. Wood[c]

Multiple imputation by chained equations is a flexible and practical approach to handling missing data. We describe the principles of the method and show how to impute categorical and quantitative variables, including skewed variables. We give guidance on how to specify the imputation model and how many imputations are needed. We describe the practical analysis of multiply imputed data, including model building and model checking. We stress the limitations of the method and discuss the possible pitfalls. We illustrate the ideas using a data set in mental health, giving Stata code fragments.

Keywords: missing data; multiple imputation; fully conditional specification

https://onlinelibrary.wiley.com/doi/full/10.1002/sim.4067

Further resources (4)

Multiple imputation with non-linear relationships:

Article

Statistical Methods in Medical Research
2015, Vol. 24(4) 462–487

Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0962280214521348
smm.sagepub.com

Multiple imputation of covariates by fully conditional specification: Accommodating the substantive model

Jonathan W Bartlett,[1] Shaun R Seaman,[2] Ian R White[2] and James R Carpenter[1,3] for the Alzheimer's Disease Neuroimaging Initiative*

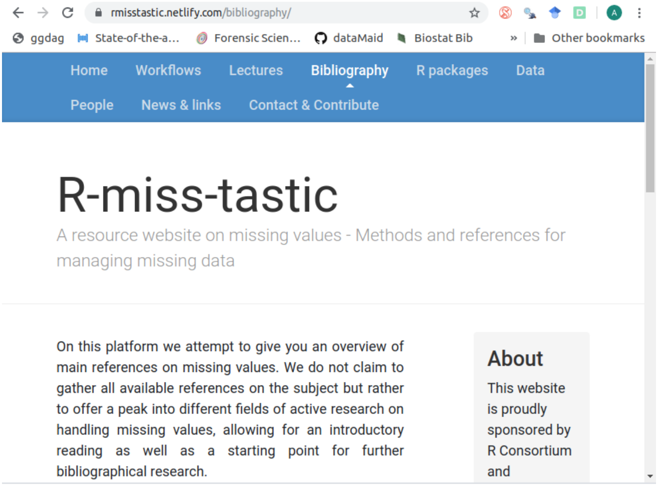
Abstract

Missing covariate data commonly occur in epidemiological and clinical research, and are often dealt with using multiple imputation. Imputation of partially observed covariates is complicated if the substantive model is non-linear (e.g. Cox proportional hazards model), or contains non-linear (e.g. squared) or interaction terms, and standard software implementations of multiple imputation may impute covariates from models that are incompatible with such substantive models. We show how imputation by fully conditional specification, a popular approach for performing multiple imputation, can be modified

https://doi.org/10.1177/0962280214521348

Further resources (5)

Website with a very thorough collection of material on missing data, emphasis on tools in R:



https://rmisstastic.netlify.app/

Comments/suggestions for this course day are very much welcome at
ahpe@sund.ku.dk