Multiple Imputation in Practice (MIMP) S28 - Thursday (v2)

https://www.gerkovink.com/mimp

Stef van Buuren, Gerko Vink, Thom Volker

July 11-14, 2022

Advanced features - MIMP M

Overview M

- Step by step
- Data operations with imputed data
- Which cells to impute?
- Group variables into blocks
- Setting up your model with formulas
- Deploy imputation model to new data
- Split training and test data
- Build your own imputation function
- Amputation: Creating missing data
- Fully and partially synthetic data

Step by step

- Calculate imputations step-by-step
- Use cases:
  - Custom convergence statistic
  - Add more iterations
- Solution: `mice.mids`

Step by step - Example

```
set.seed(12345)
mystat <- rep(NA, 5)
for (i in 1:5) {
  if (i == 1) imp <- mice(nhanes, m = 1, maxit = 1, print =
  else imp <- mice.mids(imp, maxit = 1, print = FALSE)
  mystat[i] <- cor(complete(imp)[, 2:3])[1, 2]
}
mystat
```

```
[1]  0.01083 -0.03176 -0.00714 -0.13279 -0.00143
```

Operations with imputed data

- Combine rows of `mids` objects: `rbind()`
- Combine columns of `mids` objects: `cbind()`
- Increase number of imputed datasets: `ibind()`
- Extract subset: `filter()`

Operations with imputed data - Example 1

```r
# custom imputation model by age group
grp <- nhanes$age == 1L
imp1 <- mice(nhanes[grp, -1], m = 2, print = FALSE)
```

```
Warning: Number of logged events: 1
```

```r
imp2 <- mice(nhanes[!grp, -1], m = 2, print = FALSE)
rbind(imp1, imp2)
```

```
Class: mids
Number of multiple imputations:  2
Imputation methods:
  bmi   hyp   chl
"pmm"    "" "pmm"
PredictorMatrix:
    bmi hyp chl
bmi   0   0   1
hyp   0   0   0
```

Operations with imputed data - Example 2

```r
# extract subset
imp <- mice(nhanes, m = 2, print = FALSE)
imp1 <- filter(imp, age == 1L)
nrow(complete(imp1))
```

```
[1] 12
```

Which cells to impute?

- By default, `mice` imputes the missing (`NA`) data
- The `where` argument specifies which cells are imputed
- *Overimputation*: Impute everything, create synthetic data
- *Skip imputation*: Skip imputation of selected cells (e.g. BP for the dead)
  - Warning: Unimputed missing values propagate when used as predictor
- *Monotone block imputation*:
  - Impute only cells that destroy the monotone pattern
  - Impute only cells that conform to the monotone pattern

Which cells to impute? - Example

```r
# do not impute records with age == 3
where <- make.where(nhanes)
where[nhanes$age == 3, ] <- FALSE
imp <- mice(nhanes, m = 1, where = where, print = FALSE)
md.pattern(complete(imp), plot = FALSE)
```

```
   age bmi hyp chl
21   1   1   1   1 0
2    1   1   1   0 1
1    1   0   0   1 2
1    1   0   0   0 3
     0   2   2   3 7
```

Group variables into blocks

- Hybrid models mix univariate and multivariate imputation
- `block` argument
- Examples:
  - A block of normally distributed Z-scores;
  - A set of scale items and its total score;
  - A variable with one or more transformations;
  - Two variables with one or more interaction terms;
  - Compositions that add up to a total;
  - Set of variables that are collected together.

Group variables into blocks - Example

```r
# joint multivariate normal imputation with jomoImpute
blocks <- make.blocks(nhanes, "collect")
imp <- mice(nhanes, m = 1, blocks = blocks,
            method = "jomoImpute", print = FALSE)
head(complete(imp), 3)
```

```
  age  bmi  hyp chl
1   1 24.5 1.04 170
2   2 22.7 1.00 187
3   1 32.8 1.00 187
```

Setting up your model with formulas

- Alternative to `predictorMatrix`
- Specifies imputation model by standard R syntax
- Calculates derived variables on-the-fly

Setting up your model with formulas - Example

```r
# add interaction hyp * chl to impute bmi
ft <- c("bmi ~ age + hyp * chl",
        "hyp ~ age + bmi + chl",
        "chl ~ age + bmi + hyp")
f <- name.formulas(lapply(ft, as.formula))
imp <- mice(nhanes, m = 1, formulas = f, print = FALSE)
```

Deploy imputation model to new data

- Train an imputation model on training data
- Deploy imputation model to new data
- Use `mice()` to create a `mids` object on the training data
- Use `mice.mids()` with the `newdata` argument

Deploy imputation model to new data - Example

```r
imp <- mice(nhanes, m = 1, print = FALSE)
new <- nhanes[sample(10, replace = TRUE), ]
imp.new <- mice.mids(imp, newdata = new, print = FALSE)
nrow(complete(imp.new))
```

```
[1] 10
```

Split training and test data

- New `ignore` argument since `mice 3.12.0`
- Specifies the rows to estimate the imputation model
- Allows to separate train and test data
- Prevents leakage of the test data into the imputation model

Split training and test data - Example 1

```
#' # scenario 1: train and test in the same dataset
ignore <- c(rep(FALSE, 15), rep(TRUE, 10))
imp <- mice(nhanes2, m = 1, ignore = ignore, print = FALSE)
imp.test1 <- filter(imp, ignore)
nrow(complete(imp.test1))
```

```
[1] 10
```

Split training and test data - Example 2

```
# scenario 2: train and test in separate datasets
imp.train <- mice(nhanes2[!ignore, ], m = 1, print = FALSE)
imp.test2 <- mice.mids(imp.train, newdata = nhanes2[ignore,
                       print = FALSE)
nrow(complete(imp.test2))
```

```
[1] 10
```

Build your own imputation function

- Develop dedicated imputation method
- Use cases:
    - Non-standard constraints in the data
    - Complicated if-then edits
    - Special data, e.g., text, images
- Approach:
    - Adapt related `mice.impute.xxx()` function
    - Preserve arguments `y`, `ry`, `x`, `type` and `wy`
    - Store as `mice.impute.mymeth()` in the workspace
    - Call `mice(data, method = c("pmm", "mymeth", ..., ))`
    - Optional: Store your favourites in a package

Build your own imputation function - Example 1

```r
# function in mice
mice.impute.norm <- function (y, ry, x, wy = NULL, ...)
{
    if (is.null(wy))
        wy <- !ry
    x <- cbind(1, as.matrix(x))
    parm <- .norm.draw(y, ry, x, ...)
    x[wy, ] %*% parm$beta + rnorm(sum(wy)) * parm$sigma
}
```

Build your own imputation function - Example 2

```
mice.impute.normround <- function (y, ry, x, wy = NULL, ...
{
    # round to nearest integer
    if (is.null(wy))
        wy <- !ry
    x <- cbind(1, as.matrix(x))
    parm <- .norm.draw(y, ry, x, ...)
    base::round(x[wy, ] %*% parm$beta + rnorm(sum(wy))
                * parm$sigma)
}
```

Build your own imputation function - Example 3

```
imp.norm <- mice(nhanes, method = "norm", print = FALSE)
imp.round <- mice(nhanes, method = "normround", print = FAL
head(complete(imp.norm), 3)
```

```
  age  bmi  hyp chl
1   1 25.7 1.41 129
2   2 22.7 1.00 187
3   1 28.6 1.00 187
```

```
head(complete(imp.round), 3)
```

```
  age  bmi hyp chl
1   1 25.0   1  79
2   2 22.7   1 187
3   1 25.0   1 187
```

Amputation: Creating missing data

- Amputation: inverse of imputation
- Start from a complete data matrix
- Generate missing data for simulation purposes
- Supports MCAR, MAR or MNAR missing data mechanisms
- `mice::ampute()` implements amputation
- Intended for method evaluation

Amputation: Creating missing data - Example

```r
# create missing values using defaults
cd <- cc(boys)[c("age", "hgt", "wgt")]
id <- ampute(data = cd)
md.pattern(id$amp, plot = FALSE)
```

```
    wgt hgt age
104   1   1   1   0
51    1   1   0   1
37    1   0   1   1
31    0   1   1   1
     31  37  51 119
```

Fully and partially synthetic data

Capita Selecta - MIMP O

Overview

- Skewed data
- Item imputation
- Structural equation models
- Multilevel imputation
- Vector imputation
- Compositions

Skewed data

- How to impute skewed data?
- We want to preserve skewness
- Approaches:
  - Transform to normality (log, root, Box-Cox)
  - Use robust imputation method (pmm) semi-continuous paper
  - Model skewness (ImputeRobust package) https://cran.r-project.org/web/packages/ImputeRobust/index.html

Item imputation

- Impute items from other items in same scale, include any other scale scores
- After imputation, update scale score by passive imputation
- Go to next scale

https://www.missingdata.nl/missing-data/multi-item-questionnaires/passive-imputation-example/

- If you do not yet have scales, impute all items from each other: https://stefvanbuuren.name/publications/2010%20Item%20imputation%20-%20Methodology.pdf

Impute data for structural equation models

- Find all variables that enter the structural equation model
- Create the path model (not involving latent variables)
- Impute everything from everything using default `predictorMatrix`

Imputation of multilevel data

- One of the hot spots in statistical technology
- Standard multilevel model does not deal with missing predictors
- Know the complete-data statistical analysis
- Note: Imputing the **Wide** matrix is simpler

Where are the missings?

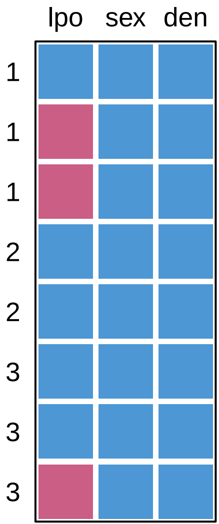
In single level data, missingness may be in the outcome and/or in the predictors

With multilevel data, missingness may be in:

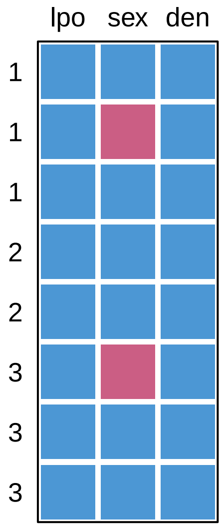
1. the outcome variable;
2. the level-1 predictors;
3. the level-2 predictors;
4. the class variable.

```
knitr::opts_chunk$set(echo = FALSE)
```

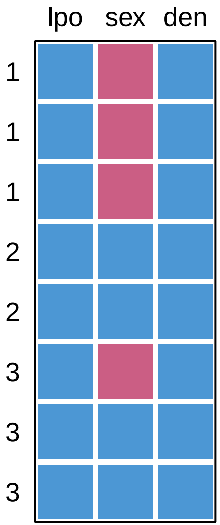
Univariate missing, level-1 outcome



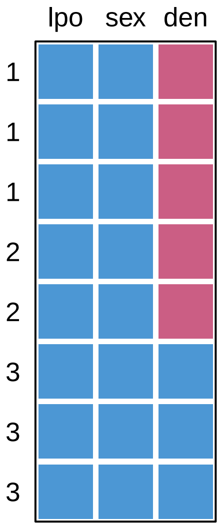
Univariate missing, level-1 predictor, sporadically missing



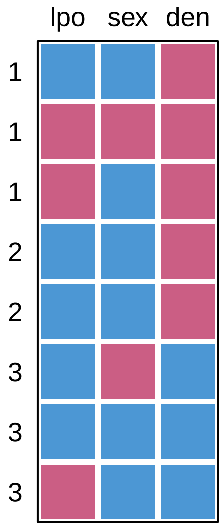
Univariate missing, level-1 predictor, systematically missing



Univariate missing, level-2 predictor



Multivariate missing



In practice: start simple, empty model

```
library(miceadds)
d <- brandsma[, c("sch", "lpo")]
pred <- make.predictorMatrix(d)
pred["lpo", "sch"] <- -2
imp <- mice(d, pred = pred, meth = "2l.pmm", m = 10,
            maxit = 1, print = FALSE, seed = 152)
```

```
* miceadds 3.13-12 (2022-05-30 15:14:07)
```

Analysis

```
library(lme4)
library(broom.mixed)
fit <- with(imp, lmer(lpo ~ (1 | sch), REML = FALSE))
summary(pool(fit))
```

```
         term estimate std.error statistic   df p.value
1 (Intercept)     40.9     0.328       125 2204       0
```

Variance components

```r
library(mitml)
testEstimates(as.mitml.result(fit),
              extra.pars = TRUE)$extra.pars
```

```
                              Estimate
Intercept~~Intercept|sch        18.462
Residual~~Residual              63.143
ICC|sch                          0.226
```

Now start adding model terms

https://stefvanbuuren.name/fimd/sec-mlguidelines.html#sec:ri1pred

Methods for multilevel imputation in `mice`

Table 7.2: Overview of methods to perform univariate multilevel imputation of continuous data. Each of the methods is available as a function called `mice.impute.[method]` in the specified `R` package.

Package	Method	Description
Continuous		
`mice`	`2l.lmer`	normal, `lmer`
`mice`	`2l.pan`	normal, `pan`
`miceadds`	`2l.continuous`	normal, `lmer` , `blme`
`micemd`	`2l.jomo`	normal, `jomo`
`micemd`	`2l.glm.norm`	normal, `lmer`
`mice`	`2l.norm`	normal, heteroscedastic
`micemd`	`2l.2stage.norm`	normal, heteroscedastic
Generic		
`miceadds`	`2l.pmm`	pmm, homoscedastic, `lmer`
`micemd`	`2l.2stage.pmm`	pmm, heteroscedastic, `mvmeta`

Methods for multilevel imputation in mice

Table 7.3: Methods to perform univariate multilevel imputation of missing discrete outcomes. Each of the methods is available as a function called `mice.impute.[method]` in the specified `R` package.

Package	Method	Description
Binary		
`mice`	`2l.bin`	logistic, `glmer`
`miceadds`	`2l.binary`	logistic, `glmer`
`micemd`	`2l.2stage.bin`	logistic, `mvmeta`
`micemd`	`2l.glm.bin`	logistic, `glmer`
Count		
`micemd`	`2l.2stage.pois`	Poisson, `mvmeta`
`micemd`	`2l.glm.pois`	Poisson, `glmer`
`countimp`	`2l.poisson`	Poisson, `glmmPQL`
`countimp`	`2l.nb2`	negative binomial, `glmmadmb`
`countimp`	`2l.zihnb`	zero-infl neg bin, `glmmadmb`

Methods for multilevel imputation in `mice`

Table 7.4: Overview of `mice.impute.[method]` functions to perform univariate multilevel imputation.

Package	Method	Description
Level-2		
`mice`	`2lonly.mean`	level-2 manifest class mean
`miceadds`	`2l.groupmean`	level-2 manifest class mean
`miceadds`	`2l.latentgroupmean`	level-2 latent class mean
`mice`	`2lonly.norm`	level-2 class normal
`mice`	`2lonly.pmm`	level-2 class pmm
`miceadds`	`2lonly.function`	level-2 class, generic
`miceadds`	`ml.lmer`	≥ 2 levels, generic

Recipe: Missing level-1

	Recipe for a level-1 target
1.	Define the most general analytic model to be applied to imputed da
2.	Select a 2l method that imputes close to the data
3.	Include all level-1 variables
4.	Include the disaggregated cluster means of all level-1 variables
5.	Include all level-1 interactions implied by the analytic model
6.	Include all level-2 predictors
7.	Include all level-2 interactions implied by the analytic model
8.	Include all cross-level interactions implied by the analytic model
9.	Include predictors related to the missingness and the target
10.	Exclude any terms involving the target

Multilevel imputation - outlook

- State of the art: https://link.springer.com/article/10.3758/s13428-020-01530-0 (May 2021)
  - introduces substantive-model-compatible model approach for multilevel data (SMC-SM)
  - especially useful when ML model contains random slopes and interactions
  - relatively simple specification of imputation model
  - R package `mdmb`
- Multilevel imputation is still very much in flux

Vector imputation

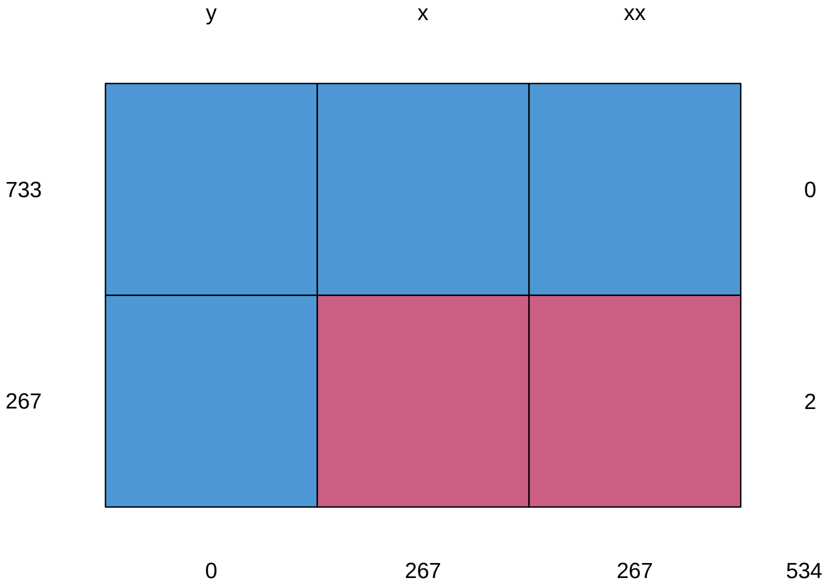
- Impute a vector instead of one value
- Use cases:
  - When data are collected in modules (matrix sampling)
  - After merge and join operations
  - Preserve related variables
- Procedure:
  - Multivariate X −> Create linear combination Xb
  - Multivariate Y −> Create linear combination Ya
  - Weights $\hat{a}$ and $\hat{b}$ maximize $\rho(Y\hat{a}, X\hat{b})$
  - Match on $X\hat{b}$ −> find 5 closest donors
  - Random donor: Take observed Y as imputation vector
- Limitation: Y's: all observed or all missing

Vector imputation - Example

```r
# Create Data
B1 <- .5
B2 <- .5
X <- rnorm(1000)
XX <- X^2
e <- rnorm(1000, 0, 1)
Y <- B1 * X + B2 * XX + e
dat <- data.frame(x = X, xx = XX, y = Y)

# Impose 25 percent MCAR Missingness
dat[0 == rbinom(1000, 1, 1 - .25), 1:2] <- NA
```

Vector imputation - Example



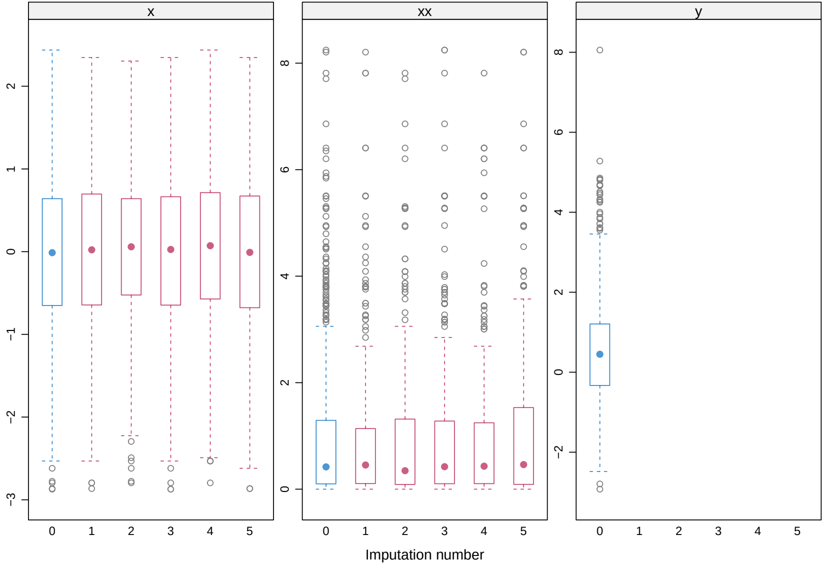
y x xx

Vector imputation - Example

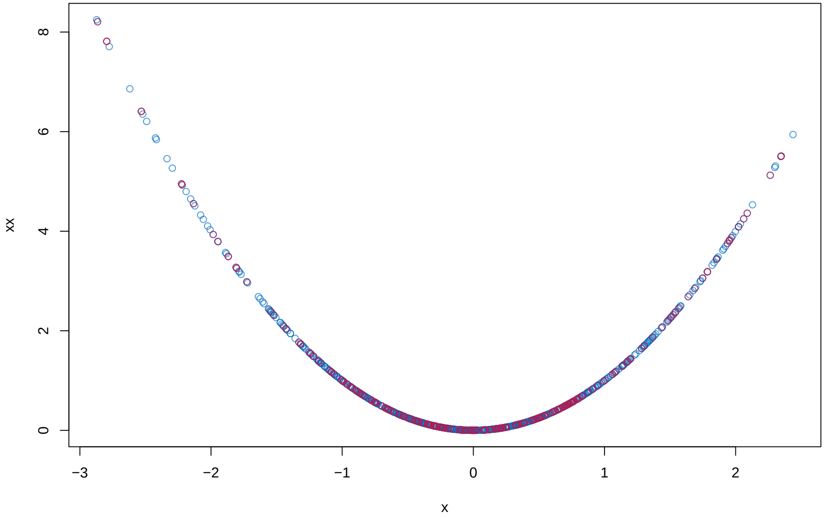
```
imp <- mice(dat, blocks = list(c("x", "xx"), "y"),
            method = c("mpmm", ""), print = FALSE)
pool(with(imp, lm(y ~ x + xx)))
```

```
Class: mipo    m = 5
         term m estimate     ubar        b        t dfcom 
1 (Intercept) 5   0.0162 0.001430 3.38e-05 0.001471   997 8
2           x 5   0.4714 0.000972 1.40e-04 0.001139   997 1
3          xx 5   0.5027 0.000497 2.14e-05 0.000522   997 6
     fmi
1 0.0300
2 0.1581
3 0.0523
```

Vector imputation - Example



Vector imputation - Example



Imputation of a composition

COMPOSITIONAL DATA

Let us consider x_0 as a combination of x_1 through x_D, such that

$$x_0 = x_1 + x_2 + ... + x_D,$$

where the integers $1, 2, ..., D$ denote the parts and the subscripted letters $x_1, ..., x_D$ denote the components.

SOME MORE BACKGROUND

All the information about compositional data is encapsulated in the ratios between the components[1]. Consequently, the proportions of the different parts of x obey

$$\frac{x_1}{x_0} + \frac{x_2}{x_0} + ... + \frac{x_D}{x_0} = 1,$$

where

$$x_1 \geq 0, x_2 \geq 0, ..., x_D \geq 0,$$

such that the sample space is defined as simplex S^D

$$S^D = \{(x_1, x_2, \ldots, x_D) : x_j \geq 0; j = 1, 2, \ldots, D; \sum_j^D x_j = x_0\}.$$

[1] Aitchison, J. (1986). *The statistical analysis of compositional data*. Chapman & Hall.

MISSINGS IN A SINGLE COMPOSITION

$$x_0 = x_1 + x_2 + x_3$$

$$x_1 + x_2 = x_0 - x_3$$

We can solve this by imputing the ratio $\pi = x_2/(x_1 + x_2)$ from a probable donor record d, yielding

$$x_1^* = \pi_{(21)}^*(x_0 - x_3),$$

and its complement

$$x_2^* = (1 - \pi_{(21)}^*)(x_0 - x_3),$$

where π_{21}^* is the imputed ratio for pair 21 and comes from the distribution

$$\Pr(\pi_{21}^* | \pi_{21}, x_0, x_3)$$

of donors with both x_2 and x_1 observed.

MULTIVARIATE MISSINGS IN A SINGLE COMPOSITION

$$x_0 = x_1 + x_2 + x_3 + x_4$$

$$x_1 + x_2 + x_3 = x_0 - x_4$$

We can solve this by finding starting values for x_1, x_2 and x_3 and iteratively updating the bivariate ratios from probable donor records.

Any starting value will be sufficient as long as the compositional structure remains intact.

All $\binom{j}{2}$ unique pairs, where j is the number of variables can be imputed by means of this approach and variables outside of the currently imputed pair, as well as the total of the composition, can serve as covariates in the prediction of π^*.

MULTIVARIATE MISSINGS IN A SINGLE COMPOSITION

$$x_0 = x_1 + x_2 + x_3 + x_4$$

The $\binom{j}{2}$ unique pairs in the above problem are

$$\begin{array}{cc} x_1 & x_2 \\ x_1 & x_3 \\ x_1 & x_4 \\ x_2 & x_3 \\ x_2 & x_4 \\ x_3 & x_4 \end{array}$$

MULTIVARIATE MISSINGS IN A SINGLE COMPOSITION

Some pairs do not need updating, as x_4 is observed and we would not want to alter observed values.

$$
\begin{array}{cc}
\color{red}{x_1} & \color{red}{x_2} \\
\color{red}{x_1} & \color{red}{x_3} \\
x_1 & x_4 \\
\color{red}{x_2} & \color{red}{x_3} \\
x_2 & x_4 \\
x_3 & x_4
\end{array}
$$

Missings in pairs where one of the variables is observed will be imputed in another pair where both variables are missing (if only one variable is missing, the equation could have been solved deductively)

Only having to consider jointly missing pairs makes PRM much faster.

PRM APPROACH

We require that starting values have been filled in and that any deductive imputation has been applied.

Carry out the following steps for all $\binom{j}{2}$ unique pairs (simplified version).

1. Calculate π^{obs} (if not defined: $\pi^{obs} = 0.5$).
2. Impute π^{mis}
3. Redistribute amounts

Repeat the above algorithm until convergence is reached. For multiple imputation do this $m \geq 2$ times

Imputations could be obtained by e.g. PMM with π^{obs} conditional on the remaining variables and the total.

x_0	x_1	x_2	x_3
32	10	15	7
18	0	9	9
22	6	3	—
14	0	—	—
8	22	—	4
30	—	—	—

BENEFITS OF PRM

By addressing the ratios between pairs, the compositional problem can be solved outside the simplex space

No need for post-hoc fixes when components are 0

The flexibility of imputation by chained equations - (m)ice

NESTED COMPOSITIONS: UNOBSERVED NESTED SUM

Suppose that we have a nested composition, where x_4 is a combination of x_5 and x_6, such that

$$x_4 = x_5 + x_6.$$

For the cases where x_4 is missing, the problem can be simplified to

$$x_1 = x_2 + x_3 + x_5 + x_6,$$

where x_4 can be deductively calculated after x_5 and x_6 are imputed.

This reduces the problem to a single composition, which can easily be solved by the proposed PRM algorithm.

NESTED COMPOSITIONS: OBSERVED NESTED SUM

For the cases where x_4 is observed, the problem can be divided into the independent imputation problems

$$x_1 = x_2 + x_3 + x_4 \quad \text{and} \quad x_4 = x_5 + x_6.$$

Solving these separate compositions is also straightforward with the proposed PRM algorithm.

In both cases donors are drawn from within the compositional level of the missing values.

MISSINGS IN MULTIPLE NESTED COMPOSITIONS

$$
\begin{array}{ccccccccccccc}
x_0 & = & x_1 & + & x_2 & + & x_3 & + & x_4 & & & & \\
 & & = & & & & & & = & & & & \\
 & & x_9 & & & & & & x_5 & & & & \\
 & & + & & & & & & + & & & & \\
 & & x_{10} & & & & & & x_6 & = & x_7 & + & x_8
\end{array}
$$

A solution for this data where x_1, x_4 and x_6 are known is simply the summation of a sumscores respective parts, such that

$$
\begin{aligned}
x_0 &= x_1 + x_2 + x_3 + x_4 \\
x_1 &= x_9 + x_{10} \\
x_4 &= x_5 + x_6 \\
x_6 &= x_7 + x_8
\end{aligned}
$$

MISSINGS IN MULTIPLE NESTED COMPOSITIONS

For unknown x_6, all components from $x_6 = x_7 + x_8$ are moved to the higher level, such that

$$\begin{aligned} x_0 &= x_1 + x_2 + x_3 + x_4 \\ x_1 &= x_9 + x_{10} \\ x_4 &= x_5 + x_7 + x_8. \end{aligned}$$

For unknown x_4 and x_6 it holds that

$$\begin{aligned} x_0 &= x_1 + x_2 + x_3 + x_5 + x_7 + x_8 \\ x_1 &= x_9 + x_{10}, \end{aligned}$$

and for unobserved x_1, x_4 and x_6 there remains one composition to be imputed, namely

$$x_0 = x_9 + x_{10} + x_2 + x_3 + x_5 + x_7 + x_8.$$

DIVIDE AND CONQUER APPROACH

Any set of nested compositions can be imputed by means of the following scheme (highly simplified!).

Start with the lowest level composition and carry out for all (nested) compositions

1. For cases where the sumscore of a nested composition is missing
   1.1 Promote the missingness problem to the next higher level composition and save it for later.
2. For cases where the sumscore of a nested composition is observed
   2.1 Impute the missing parts in the composition by means of PRM.
   2.2 Calculate unobserved nested totals (if any) in the current composition based on the imputed parts.

Repeat the above until convergence is reached. For multiple imputation do this $m \geq 2$ times

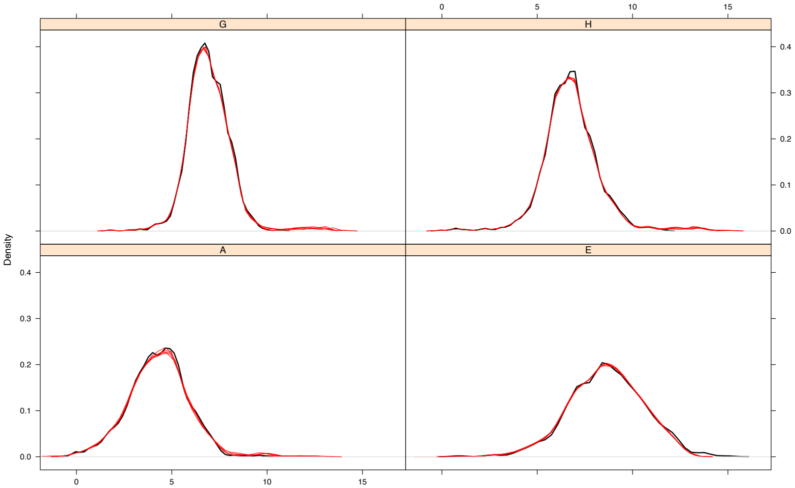
EVALUATING PRM

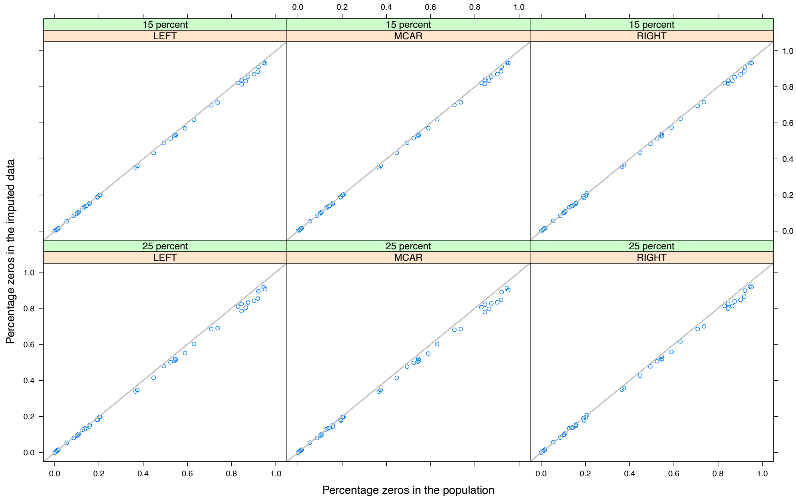
Simulation study: Dutch Wholesaler Data (DWD) for 2007. The DWD dataset contains edited information on 1067 wholesalers for a set of cost statistics (a, e, g and h) that sum op to a set total x_0, leading to composition

$$x_0 = a + e + g + h,$$

x_0	the total operating costs		
a	company depreciation	single measure	
e	buying costs	5 parts	
g	personnel costs	9 parts	
h	other costs	21 parts	$h_1 = h_2 + h_3 + h_4$

3 missingness mechanisms: left-tailed MAR, right-MAR and MCAR


G
H
A
E
Density
0.0
0.1
0.2
0.3
0.4
0
5
10
15


15 percent
LEFT
15 percent
MCAR
15 percent
RIGHT
25 percent
LEFT
25 percent
MCAR
25 percent
RIGHT
0.0
0.2
0.4
0.6
0.8
1.0
Percentage zeros in the imputed data
Percentage zeros in the population