

A simple example of multiple imputation using the mice package

Brice Ozenne

March 8, 2021

This document gathers code from the documentation of the mice package. See <https://stefvanbuuren.name/mice/>.

Load packages

```
library(lava)
library(mice)
library(data.table)
library(ggplot2)
```

1 Simulate data

Just to have an example to work with. Generative model:

```
mSim <- lvm(Y~group+season+bmi+gender+age)
categorical(mSim, labels = c("winter", "summer")) <- ~season
categorical(mSim, labels = c("SAD", "HC")) <- ~group
categorical(mSim, labels = c("Male", "Female")) <- ~gender
distribution(mSim, ~bmi) <- lava::gaussian.lvm(mean = 22, sd = 3)
distribution(mSim, ~age) <- lava::uniform.lvm(20, 80)
```

Sampling

```
n <- 1e2
set.seed(10)
dt.data <- as.data.table(sim(mSim, n))
```

Add missing values

```
dt.data[1:10, bmi:=NA]
```

2 Working with mice

2.1 Step 1: Inspect the missing data pattern

Check the number of missing values in the dataset:

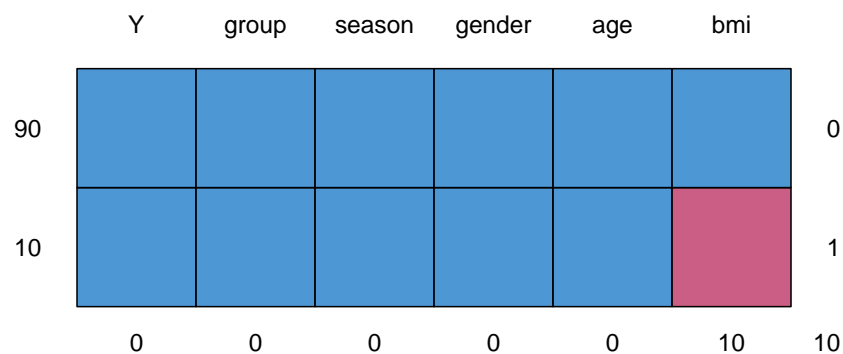
```
colSums(is.na(dt.data))
```

```
Y group season bmi gender age
0      0      0    10      0      0
```

Missing data patterns:

```
md.pattern(dt.data)
```

```
Y group season gender age bmi
90 1      1      1      1  1  1  0
10 1      1      1      1  1  0  1
   0      0      0      0  0  0 10 10
```



Note: with more patterns it becomes clear that the left column counts the number of observations with a specific pattern. The right column display the index of the pattern. The bottom row counts the number of missing value relative to each variable (and the total number of missing values).

```
dt.data2 <- copy(dt.data)
dt.data2[5:10,age:=NA]
md.pattern(dt.data2)
```

```
Y group season gender age bmi
90 1      1      1      1  1  1  0
4  1      1      1      1  1  0  1
6  1      1      1      1  0  0  2
   0      0      0      0  6 10 16
```

2.2 Step 2: Define imputation model

```
all.variables <- c("Y","group","season","bmi","gender","age")
n.variables <- length(all.variables)
Mlink <- matrix(0, n.variables, n.variables,
  dimnames = list(all.variables,all.variables))
Mlink["bmi",c("group","season","gender","age")] <- 1
Mlink
```

	Y	group	season	bmi	gender	age
Y	0	0	0	0	0	0
group	0	0	0	0	0	0
season	0	0	0	0	0	0
bmi	0	1	1	0	1	1
gender	0	0	0	0	0	0
age	0	0	0	0	0	0

A value of 1 means that the column variable is used as a predictor for the target block (in the rows).

2.3 Step 3: Generate imputed datasets

Generate imputed values

```
n.imputed <- 3 ## number of imputed datasets
dt.mice <- mice(dt.data,
  m=n.imputed,
  maxit = 50, # number of iterations to obtain the imputed dataset
  predictorMatrix = Mlink,
  method = 'pmm', # Predictive mean matching, only ok for continuous
    variables, it is possible to set constrains for positive variables
  seed = 500, printFlag = FALSE)
summary(dt.mice)
```

Class: mids

Number of multiple imputations: 3

Imputation methods:

Y	group	season	bmi	gender	age
""	""	""	"pmm"	""	""

PredictorMatrix:

	Y	group	season	bmi	gender	age
Y	0	0	0	0	0	0
group	0	0	0	0	0	0
season	0	0	0	0	0	0
bmi	0	1	1	0	1	1
gender	0	0	0	0	0	0
age	0	0	0	0	0	0

2.3.1 Interacting with the mice object

Missingness indicator:

```
table(cci(dt.mice))
```

FALSE	TRUE
10	90

Complete case dataset:

```
str(cc(dt.mice))
```

```
'data.frame':      90 obs. of  6 variables:
 $ Y      : num  102 82.9 94.9 61.4 82 ...
 $ group  : Factor w/ 2 levels "SAD","HC": 1 2 1 1 2 2 2 2 2 2 ...
```

```
$ season: Factor w/ 2 levels "winter","summer": 2 2 1 1 2 1 2 1 1 1 ...
$ bmi    : num  21.6 21.9 20.6 26.4 28.5 ...
$ gender: Factor w/ 2 levels "Male","Female": 2 2 2 1 1 2 2 1 2 2 ...
$ age    : num  77.3 57.2 73.6 34 50.7 ...
```

Extract observations with missing values:

```
str(ic(dt.mice))
```

```
'data.frame':      10 obs. of  6 variables:
 $ Y      : num  92.2 91 63.1 76.2 75.2 ...
 $ group  : Factor w/ 2 levels "SAD","HC": 1 2 1 2 1 2 2 1 2 2
 $ season: Factor w/ 2 levels "winter","summer": 2 2 2 2 1 2 2 1 1 1
 $ bmi    : num  NA NA NA NA NA NA NA NA NA NA
 $ gender: Factor w/ 2 levels "Male","Female": 2 2 1 2 2 2 1 2 1 1
 $ age    : num  68.5 62.6 34.9 50.6 54.1 ...
```

Dataset after multiple imputation:

```
str(complete(dt.mice, action = 1)) ## first imputed dataset
```

```
'data.frame':      100 obs. of  6 variables:
 $ Y      : num  92.2 91 63.1 76.2 75.2 ...
 $ group  : Factor w/ 2 levels "SAD","HC": 1 2 1 2 1 2 2 1 2 2 ...
 $ season: Factor w/ 2 levels "winter","summer": 2 2 2 2 1 2 2 1 1 1 ...
 $ bmi    : num  25.9 21 16.7 20.2 25.9 ...
 $ gender: Factor w/ 2 levels "Male","Female": 2 2 1 2 2 2 1 2 1 1 ...
 $ age    : num  68.5 62.6 34.9 50.6 54.1 ...
```

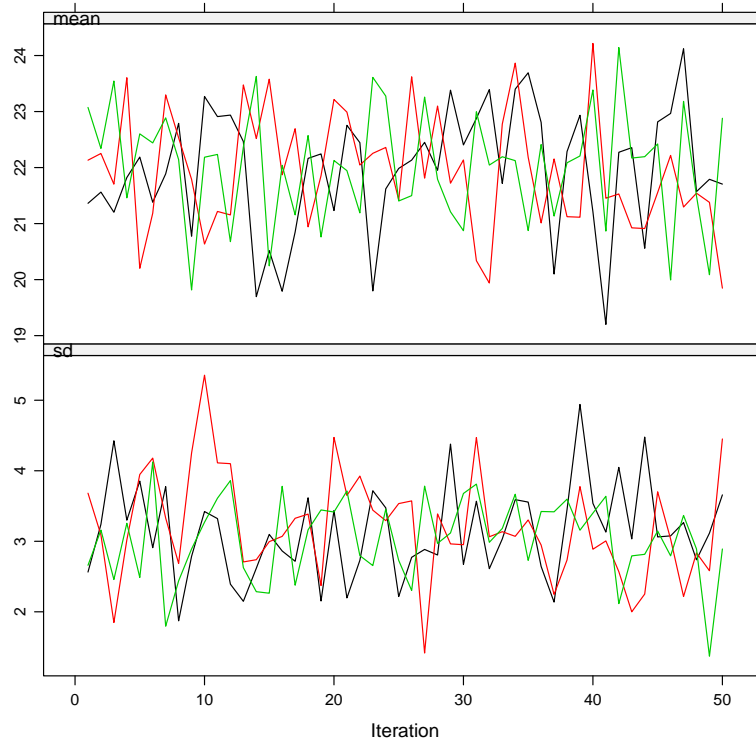
```
str(complete(dt.mice, action = 2)) ## second imputed dataset
```

```
'data.frame':      100 obs. of  6 variables:
 $ Y      : num  92.2 91 63.1 76.2 75.2 ...
 $ group  : Factor w/ 2 levels "SAD","HC": 1 2 1 2 1 2 2 1 2 2 ...
 $ season: Factor w/ 2 levels "winter","summer": 2 2 2 2 1 2 2 1 1 1 ...
 $ bmi    : num  17.5 18.1 24.3 20.7 20.9 ...
 $ gender: Factor w/ 2 levels "Male","Female": 2 2 1 2 2 2 1 2 1 1 ...
 $ age    : num  68.5 62.6 34.9 50.6 54.1 ...
```

2.4 Step 4: Check the imputed datasets

2.4.1 Convergence of the imputation algorithm

```
plot(dt.mice, layout = c(1,2))
```



2.4.2 Visualizing the imputed values

Visualize imputed value values and check they are plausible (e.g. mice is not imputed a BMI of 75):

```
dt.mice$imp$bmi
```

	1	2	3
1	25.86170	17.51962	26.03528
2	20.97076	18.12224	21.60139
3	16.68404	24.25679	17.86661
4	20.15124	20.71567	26.03528
5	25.86170	20.92015	25.24236
6	24.80171	25.02219	20.62111
7	21.42336	25.02219	25.02219
8	21.00639	20.90548	20.15124
9	24.76365	12.99571	24.80171
10	15.52519	12.99571	21.42336

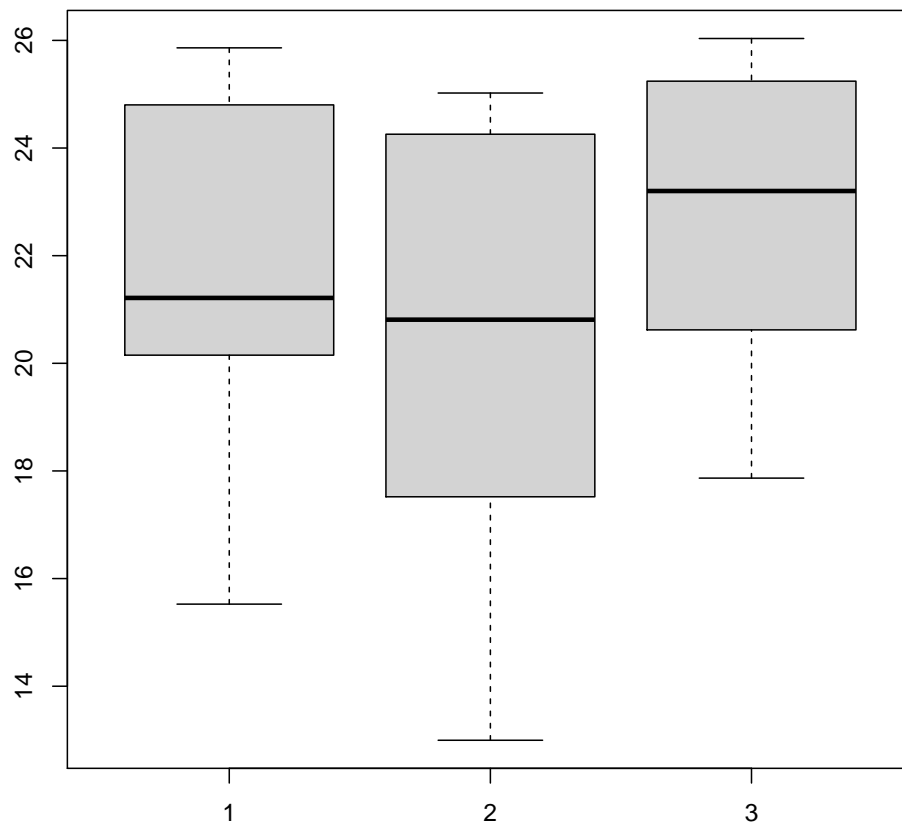
The rows correspond to the 3 different imputed datasets and the columns to 10 imputed values per dataset. One can also summarize the imputed values computing their quantiles:

```
apply(dt.mice$imp$bmi,2,quantile)
```

	1	2	3
0%	15.52519	12.99571	17.86661
25%	20.35612	17.67028	20.82167
50%	21.21487	20.81058	23.20155
75%	24.79219	23.42263	25.18731
100%	25.86170	25.02219	26.03528

Boxplot of the imputed values:

```
boxplot(dt.mice$imp$bmi)
```



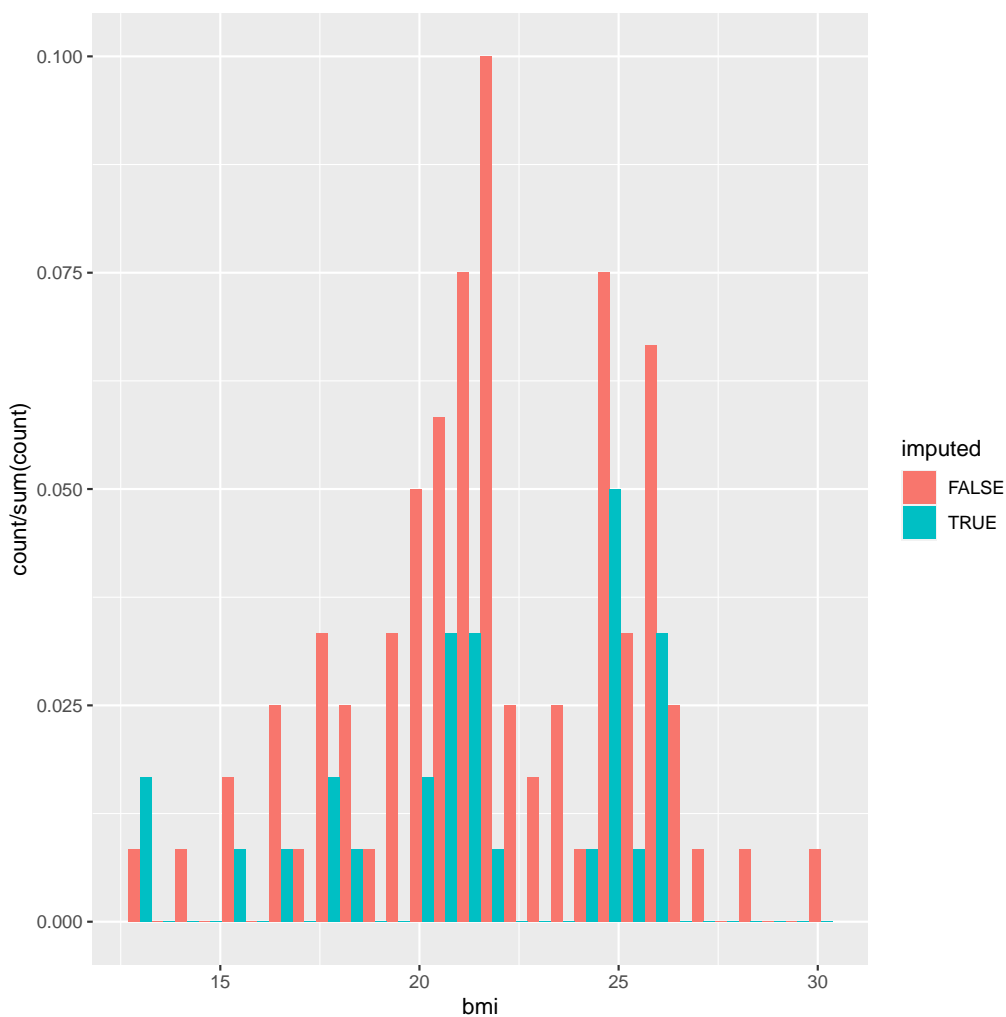
Imputed values vs. observed values

```
dt.bmi <- rbind(data.table(bmi = unlist(dt.mice$imp$bmi), imputed = TRUE),  
  data.table(bmi = na.omit(dt.data$bmi), imputed = FALSE))
```

Histogram

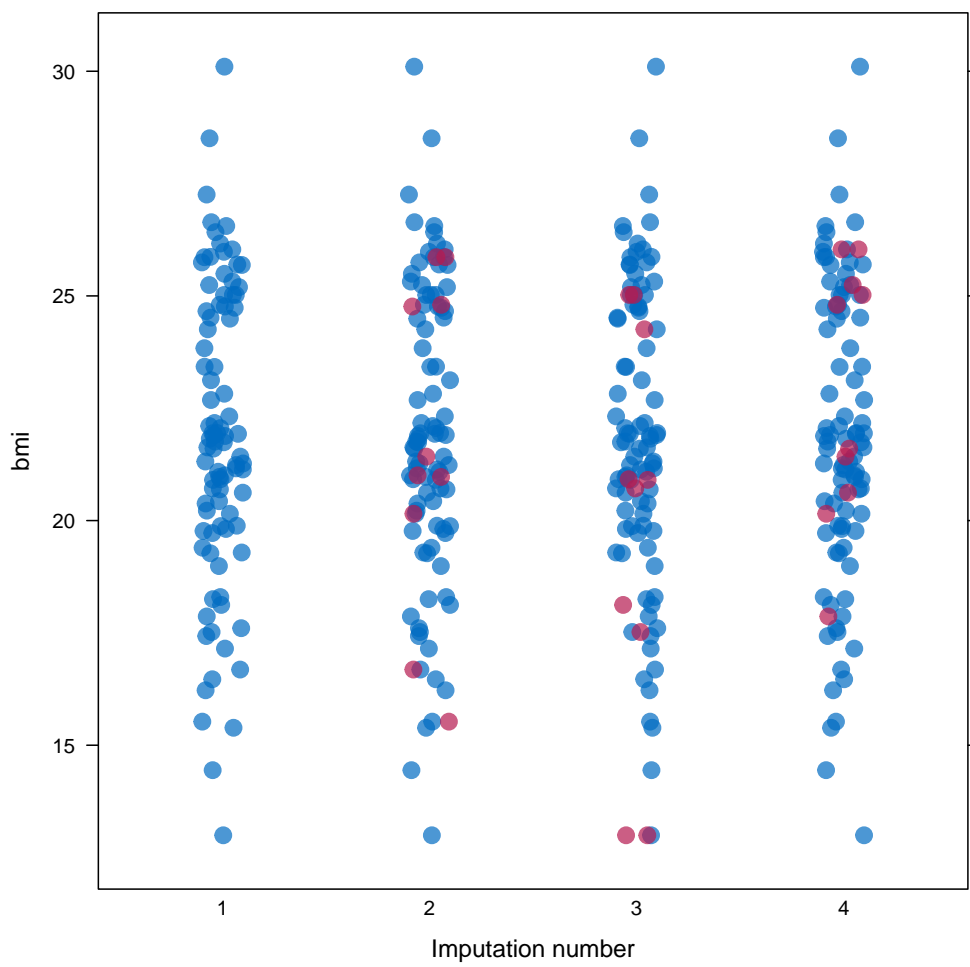
```
gg1.bmi <- ggplot(dt.bmi, aes(bmi, group = imputed, fill = imputed))  
gg1.bmi <- gg1.bmi + geom_histogram(aes(y=..count../sum(..count..)),  
  position = "dodge")  
gg1.bmi
```

'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.



One more plot:

```
stripplot(dt.mice, bmi~.imp, pch=20, cex=2)
```



2.5 Step 3: Fit the statical model on each imputed dataset

```
e.mice <- with(data = dt.mice,  
              lm(Y~group+season+bmi+gender+age)  
              )  
e.mice
```

```
call :  
with.mids(data = dt.mice, expr = lm(Y ~ group + season + bmi +  
  gender + age))  
  
call1 :  
mice(data = dt.data, m = n.imputed, method = "pmm", predictorMatrix = Mlink,  
      maxit = 50, printFlag = FALSE, seed = 500)
```

```
nmis :  
      Y group season    bmi gender    age  
      0     0      0     10      0      0
```

```
analyses :  
[[1]]
```

```
Call:  
lm(formula = Y ~ group + season + bmi + gender + age)
```

```
Coefficients:  
(Intercept)      groupHC  seasonsummer          bmi  genderFemale          age  
      2.8846       0.7238       1.2174       0.8602       0.5507       1.0058
```

```
[[2]]
```

```
Call:  
lm(formula = Y ~ group + season + bmi + gender + age)
```

```
Coefficients:  
(Intercept)      groupHC  seasonsummer          bmi  genderFemale          age  
      3.8365       0.6732       0.8917       0.7907       0.7422       1.0221
```

```
[[3]]
```

```
Call:  
lm(formula = Y ~ group + season + bmi + gender + age)
```

Coefficients:

(Intercept)	groupHC	seasonsummer	bmi	genderFemale	age
2.2639	0.5601	1.1779	0.8919	0.7209	1.0025

We can check that using `with` is equivalent to run the linear regression on the imputed dataset:

```
dt.tempo <- copy(dt.data)
dt.tempo[is.na(bmi), bmi := dt.mice$imp$bmi[,1]]
lm(Y ~ group + season + bmi + gender + age, data = dt.tempo)
```

Call:

```
lm(formula = Y ~ group + season + bmi + gender + age, data = dt.tempo)
```

Coefficients:

(Intercept)	groupHC	seasonsummer	bmi	genderFemale	age
2.8846	0.7238	1.2174	0.8602	0.5507	1.0058

2.6 Step 4: Pool the results over the imputed datasets

```
ePool.mice <- pool(e.mice)
summary(ePool.mice)
```

	term	estimate	std.error	statistic	df	p.value
1	(Intercept)	2.9949755	1.73919838	1.722044	18.777429	1.014936e-01
2	groupHC	0.6523698	0.42434760	1.537348	78.067966	1.282516e-01
3	seasonsummer	1.0956572	0.46036133	2.379994	30.027026	2.386256e-02
4	bmi	0.8475895	0.08552179	9.910802	7.101823	2.060982e-05
5	genderFemale	0.6712684	0.42707145	1.571794	66.444876	1.207484e-01
6	age	1.0101585	0.01664369	60.693193	6.176972	8.221885e-10

The (pooled) estimate is the average of the estimates relative to each imputed dataset:

```
Q.coef <- colMeans(do.call(rbind,lapply(e.mice$analyses, coef)))
Q.coef
```

(Intercept)	groupHC	seasonsummer	bmi	genderFemale	age
2.9949755	0.6523698	1.0956572	0.8475895	0.6712684	1.0101585

The variance is a bit more complex and involves:

- the within-imputation variance (depends on the sample size)

```
covW <- Reduce("+",lapply(e.mice$analyses, vcov))/n.imputed
print(covW, digits = 5)
```

	(Intercept)	groupHC	seasonsummer	bmi	genderFemale	age
(Intercept)	2.1882899	-0.13040004	-0.04842230	-7.9278e-02	-0.10952359	-5.2588e-03
groupHC	-0.1304000	0.17070925	0.01361091	3.0162e-03	0.00655971	-6.6906e-04
seasonsummer	-0.0484223	0.01361091	0.16981498	-2.3136e-03	0.01559623	-2.0107e-04
bmi	-0.0792775	0.00301620	-0.00231365	3.7376e-03	-0.00042973	-4.6547e-05
genderFemale	-0.1095236	0.00655971	0.01559623	-4.2973e-04	0.16771404	2.9411e-04
age	-0.0052588	-0.00066906	-0.00020107	-4.6547e-05	0.00029411	1.3100e-04

- the between-imputation variance (depends on the amount of missing data)

```
ls.diffCoef <- lapply(e.mice$analyses, function(iI){coef(iI)-Q.coef})
covB <- Reduce("+",lapply(ls.diffCoef,tcrossprod))/(n.imputed-1)
print(covB, digits = 5)
```

```

      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 0.6273908 0.03854326 -0.1225940 -0.04085088 0.01833254 0.00804290
[2,] 0.0385433 0.00702122 -0.0015717 -0.00218666 -0.00585420 0.00032147
[3,] -0.1225940 -0.00157170 0.0315882 0.00839604 -0.01252242 -0.00179271
[4,] -0.0408509 -0.00218666 0.0083960 0.00268231 -0.00167818 -0.00053567
[5,] 0.0183325 -0.00585420 -0.0125224 -0.00167818 0.01100699 0.00049399
[6,] 0.0080429 0.00032147 -0.0017927 -0.00053567 0.00049399 0.00010951

```

- the simulation error

```

covE <- covB/n.imputed
print(covE, digits = 5)

```

```

      [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,] 0.2091303 0.01284775 -0.04086467 -0.01361696 0.00611085 2.6810e-03
[2,] 0.0128478 0.00234041 -0.00052390 -0.00072889 -0.00195140 1.0716e-04
[3,] -0.0408647 -0.00052390 0.01052939 0.00279868 -0.00417414 -5.9757e-04
[4,] -0.0136170 -0.00072889 0.00279868 0.00089410 -0.00055939 -1.7856e-04
[5,] 0.0061108 -0.00195140 -0.00417414 -0.00055939 0.00366900 1.6466e-04
[6,] 0.0026810 0.00010716 -0.00059757 -0.00017856 0.00016466 3.6504e-05

```

The total variance is:

```

covT <- covW + covB + covE

```

leading to the standard errors:

```

sqrt(diag(covT))

```

```

(Intercept)      groupHC seasonsummer      bmi genderFemale      age
1.73919838      0.42434760      0.46036133      0.08552179      0.42707145      0.01664369

```

There is also a function to extract the R-squared:

```

pool.r.squared(e.mice)

```

```

      est      lo 95      hi 95 fmi
R^2 0.9890535 0.9819615 0.9933666 NaN

```

```

vec.rsquared <- sapply(e.mice$analyses, function(iImp){
  summary(iImp)$r.squared
})
tanh(mean(atanh(vec.rsquared)))

```

```

[1] 0.9890535

```

3 Special case: imputation using a specific law and no covariate

Mice can be adapted in order, for instance, to sample from a uniform distribution or a truncated normal distribution. First define a function able to generate data like:

```
mice.impute.SI_unif <- function(y, ry, ...){ ## uniform law
  n.NA <- sum(ry==FALSE)
  sample <- runif(n.NA, min = 0, max = 1)
  return(cbind(sample))
}
```

or

```
mice.impute.SI_tnorm <- function(y, ry, ...){ ## truncated normal law
  require(truncnorm)
  n.NA <- sum(ry==FALSE)
  sample <- rtruncnorm(n.NA, a = 0, b = 1, mean = 1, sd = 0.1)
  return(cbind(sample))
}
```

Then prepare the matrix indicating which variable should be used during the imputation:

```
impute.var <- c("bmi", "group")
Mlink2 <- matrix(0,
  nrow = length(impute.var),
  ncol = length(impute.var),
  dimnames = list(impute.var, impute.var))
Mlink2["bmi", "group"] <- 1
Mlink2
```

```
      bmi group
bmi    0    1
group  0    0
```

Then run mice as usual except that the method should correspond to one of the previous functions:

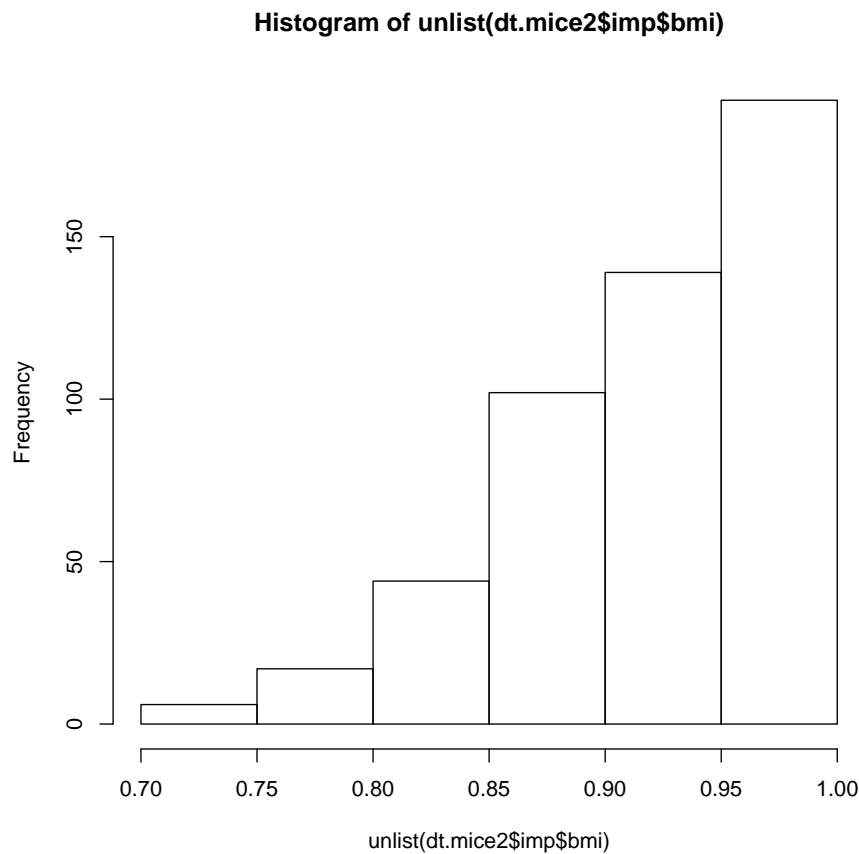
```
n.imputed <- 50 ## number of imputed datasets
set.seed(1)
dt.mice2 <- mice(dt.data,
  m=n.imputed,
  maxit = 1, # not relevant
  predictorMatrix = Mlink2, # not relevant
  method = 'SI_tnorm', # function previous define (without "mice.impute
  ".")
  seed = 500, printFlag = FALSE)
```

Then as usual one should check that the imputed values are satisfying:

```
quantile(unlist(dt.mice2$imp$bmi))
```

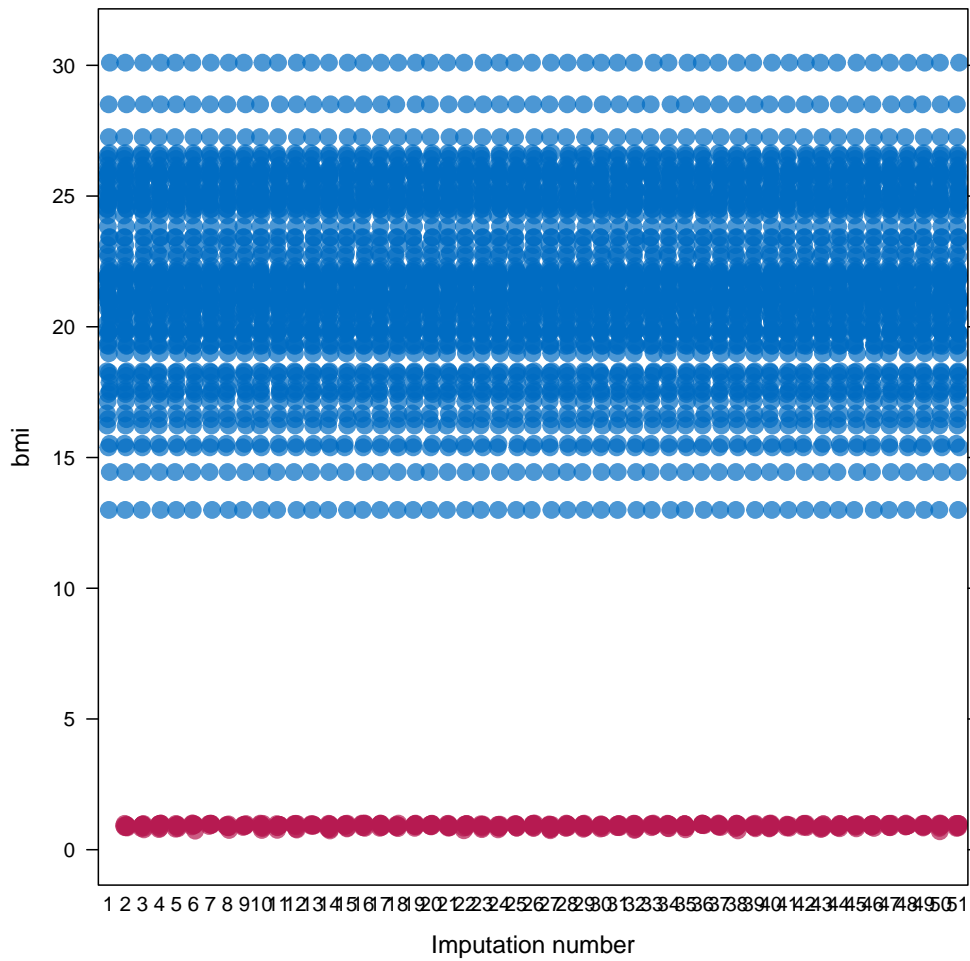
```
      0%      25%      50%      75%     100%
0.7041556 0.8790477 0.9317021 0.9687630 0.9997288
```

```
hist(unlist(dt.mice2$imp$bmi))
```



One more plot:

```
stripplot(dt.mice2, bmi~.imp, pch=20, cex=2)
```



Here for instance the imputed values does not overlap the observed one so something (i.e. the parameters of the distribution used for the imputation) is wrong.

4 Reporting guideline

From <https://stefvanbuuren.name/Winnipeg/Lectures/Winnipeg.pdf>:

- Amount of missing data
- Reasons for missingness
- Differences between complete and incomplete data
- Method used to account for missing data
- Software
- Number of imputed datasets
- Imputation model
- Derived variables
- Diagnostics
- Pooling
- Listwise deletion
- Sensitivity analysis