Imputing missing data using R

Rmanchester.org

June 2, 2015

Graeme Hutcheson

Manchester Institute of Education
University of Manchester

This session aims to provide an introduction to imputing missing data using tools available in R.

This demonstration uses 'real' data from a project investigating mathematics teaching in schools. The analysis was particularly challenging, as there were substantial amounts of missing data and relatively small expected effect sizes.

One of the variables in the study had substantial missing data, but this data became available at a later date. This made it possible to compare an analysis based on the original data (n=495) with an analysis based on more complete data (n=1374).

This paper evaluates the feasibility of imputing the missing data, even though the original data was from a poorly-sampled binary categorical variable.

Considerations for imputing data

The imputation technique needs to be accessible to PhD students (many techniques are difficult and computationally intensive).

The technique should be available using generally-accessible software.

If possible, a graphical-user-interface should be available.

The data set

This paper models whether or not a pupil 'drops-out' of a mathematics course based on the course they are taking (AS-traditional or Uses of Maths), their previous GCSE-grade, their disposition to study and their rated self-efficacy with mathematics.

A main-effects binary logit model was chosen to illustrate the imputation procedure...

$$\text{Drop-out} \sim \text{course} + \text{GCSE} + \text{disposition} + \text{efficacy}$$

Information about drop-out

Information about drop-out was available for 495 children originally.

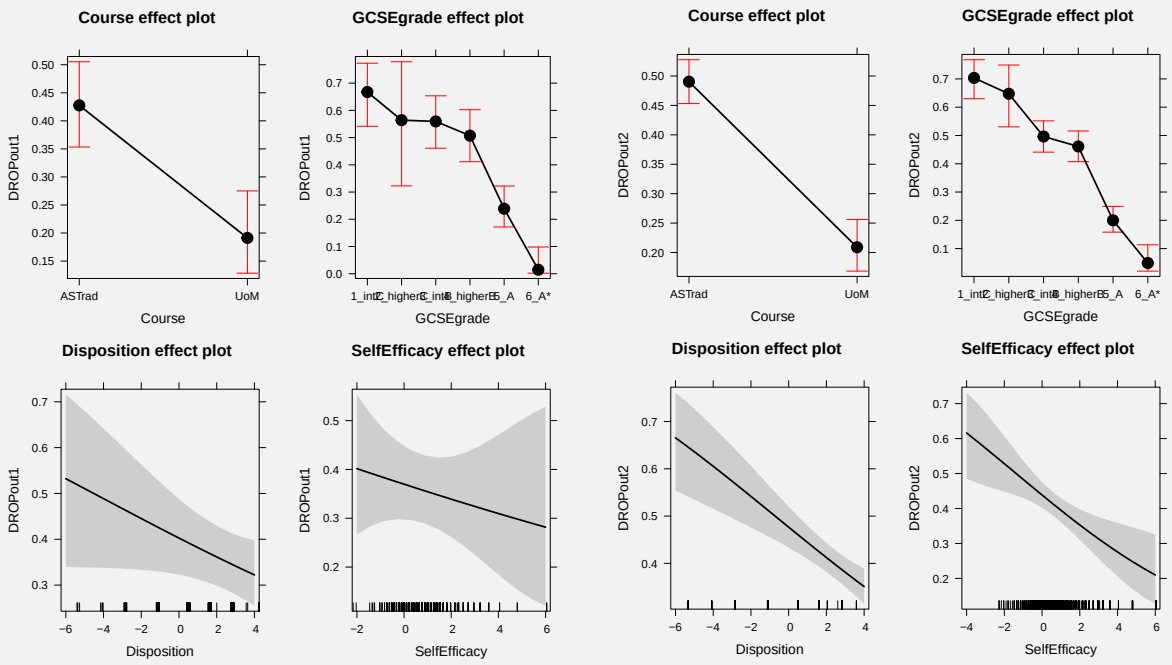
At a later date, another data base provided information about drop-out for an extra 879 cases (increasing the sample size to 1374).

Did the extra data make any difference to the analysis?

Could these missing data have been imputed from the original data?

Models of Dropout using original and retrieved data

Original Data (n=459) Retrieved Data (n=1374)



Modelling Dropout using original data

```
glm(formula = DROPout1 ~ Course + GCSEgrade + Disposition +
              Disposition + SelfEfficacy,
    family = binomial(logit)Disposition +, data = ALLdata)
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.23719	0.31868	3.882	0.000103
CourseUoM	-1.15016	0.25838	-4.451	8.53e-06
GCSEgrade2_higherC	-0.43881	0.56797	-0.773	0.439768
GCSEgrade3_intB	-0.45760	0.32219	-1.420	0.155534
GCSEgrade4_higherB	-0.66722	0.34204	-1.951	0.051093
GCSEgrade5_A	-1.85488	0.36567	-5.073	3.92e-07
GCSEgrade6_A*	-4.89948	1.05788	-4.631	3.63e-06
Disposition	-0.08730	0.04641	-1.881	0.059931
SelfEfficacy	-0.06709	0.09926	-0.676	0.499091

Modelling Dropout using retrieved data

```
glm(formula = DROPout2 ~ Course + GCSEgrade + Disposition +
              SelfEfficacy,
              family = binomial(logit), data = ALLdata)
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.61694	0.20226	7.995	1.30e-15
CourseUoM	-1.29311	0.15936	-8.114	4.88e-16
GCSEgrade2_higherC	-0.25615	0.28919	-0.886	0.37576
GCSEgrade3_intB	-0.87884	0.20043	-4.385	1.16e-05
GCSEgrade4_higherB	-1.01988	0.20984	-4.860	1.17e-06
GCSEgrade5_A	-2.25137	0.23560	-9.556	< 2e-16
GCSEgrade6_A*	-3.83002	0.50079	-7.648	2.04e-14
Disposition	-0.13037	0.02857	-4.564	5.03e-06
SelfEfficacy	-0.17994	0.05622	-3.200	0.00137

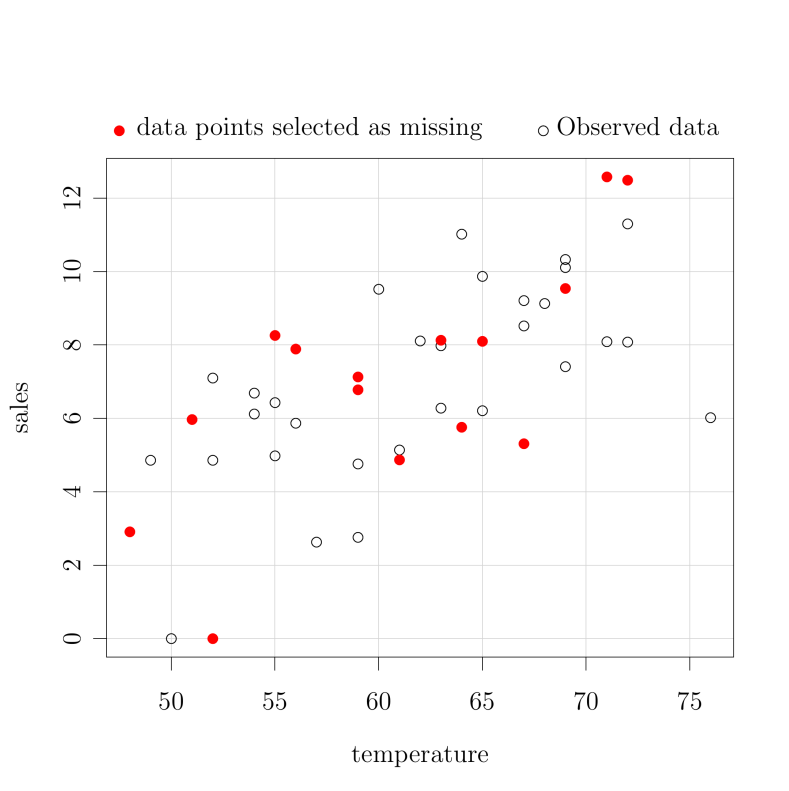
Comparing models using original and retrieved data

Although similar patterns are provided by both models (see effect displays), there are some important differences when interpreting the significance estimates. Most notably, the significance of Disposition and Self-efficacy and the GCSE-B-grades.

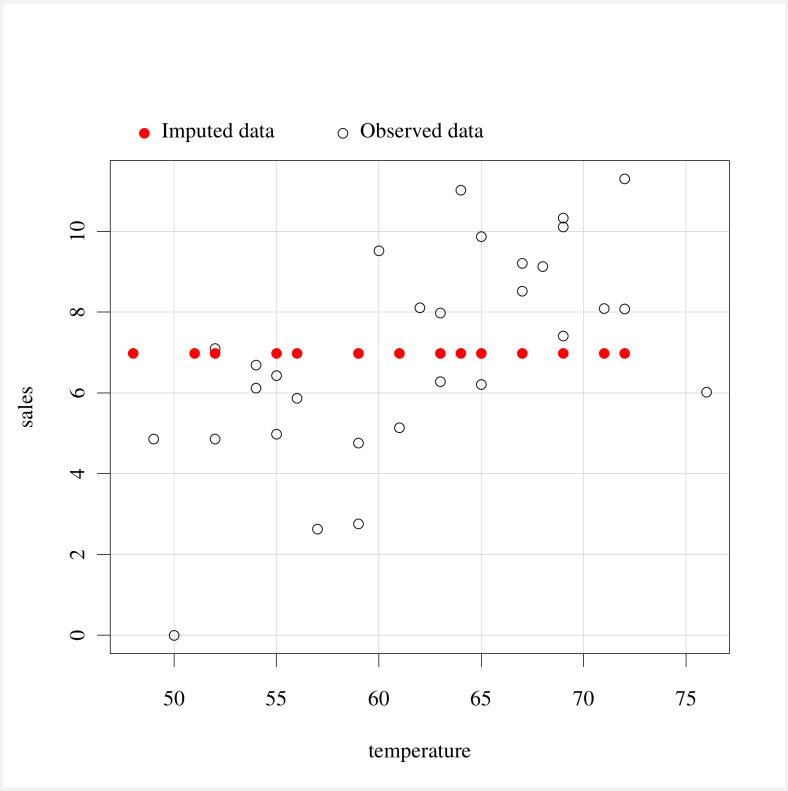
For this model, missing data is an issue.

A number of simple methods for replacing the missing values are commonly used, but are very poor...

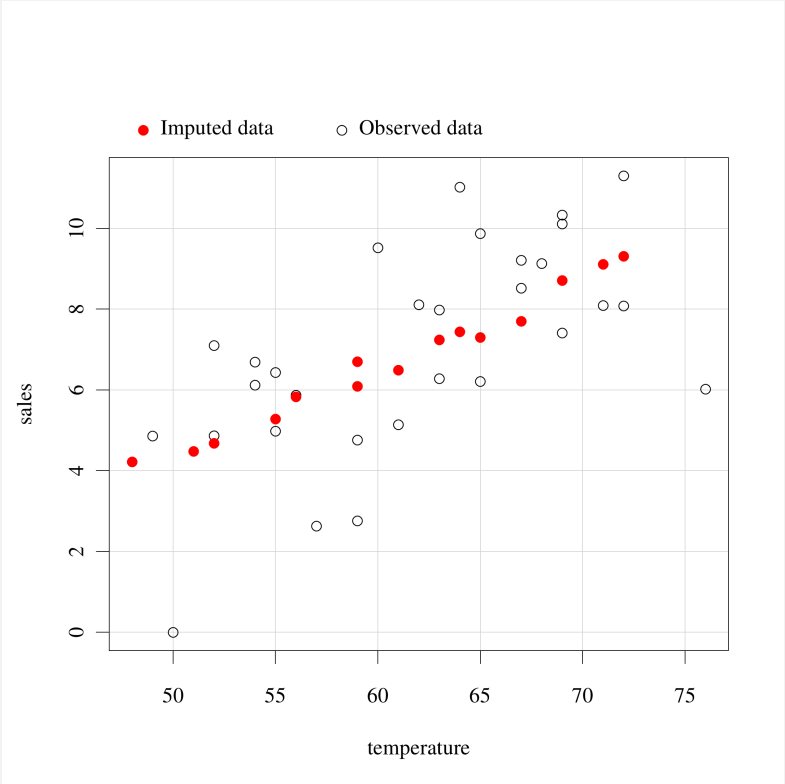
Selecting missing from a simple dataset



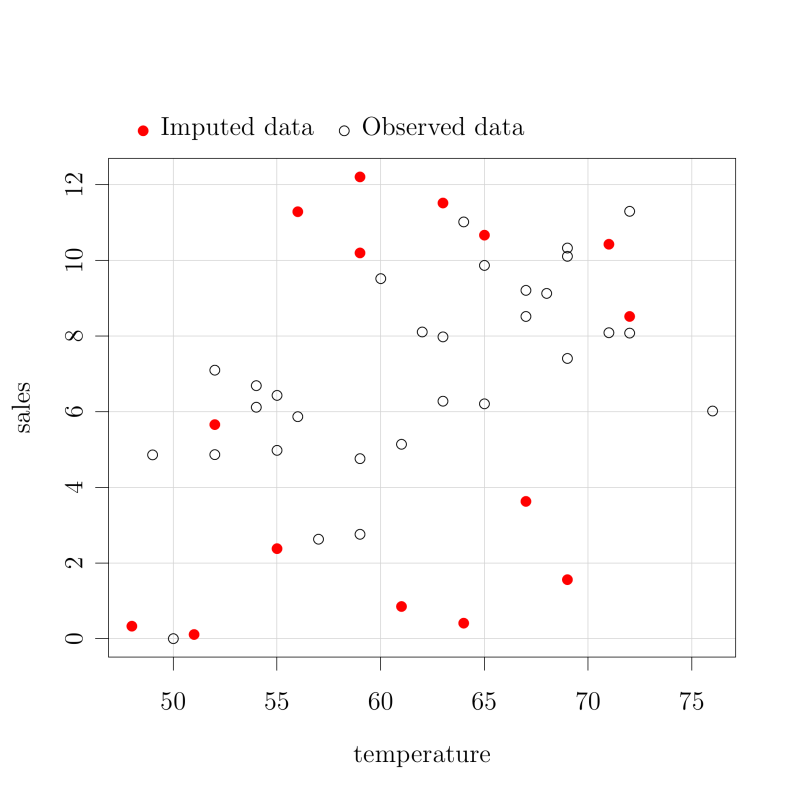
replacing missing values with the mean



replacing missing values with predictions from regression



replacing missing values with random values



From the graphs above, it is clear that the imputed data do not have the same distribution as the sample from which they are drawn.

'Simple' methods for replacing missing data are inadequate, but are still 'probably' better than simply ignoring the missing values (as happens in complete-data techniques such as step-wise regression).

What is needed is a technique that reproduces the missing data so that it more accurately reflects the sample it was taken from.

There are many packages in R that utilise a variety of imputation techniques...

Missing data analysis in R

- Amelia II: A Program for Missing Data using multiple imputation.
- arrayImpute: Missing data imputation for microarray data
- cat: Analysis of categorical-variable datasets with missing values
- EMV: Estimation of Missing Values for a Data Matrix
- impute: Imputation for microarray data
- mi: Missing Data Imputation and Model Checking
- mice: Multivariate Imputation by Chained Equations

Missing data analysis in R

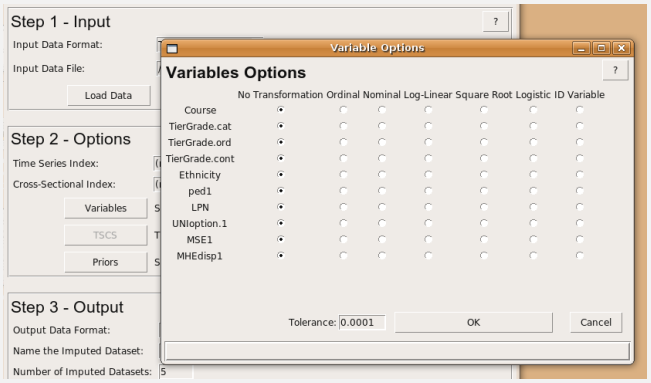
- mirf: Multiple imputation and random forests for unobservable phase, high-dimensional data.
- mitools: Tools for multiple imputation of missing data
- mix: Estimation/multiple Imputation for Mixed Categorical and Continuous Data
- pan: Multiple imputation for multivariate panel or clustered data

Missing data analysis in R

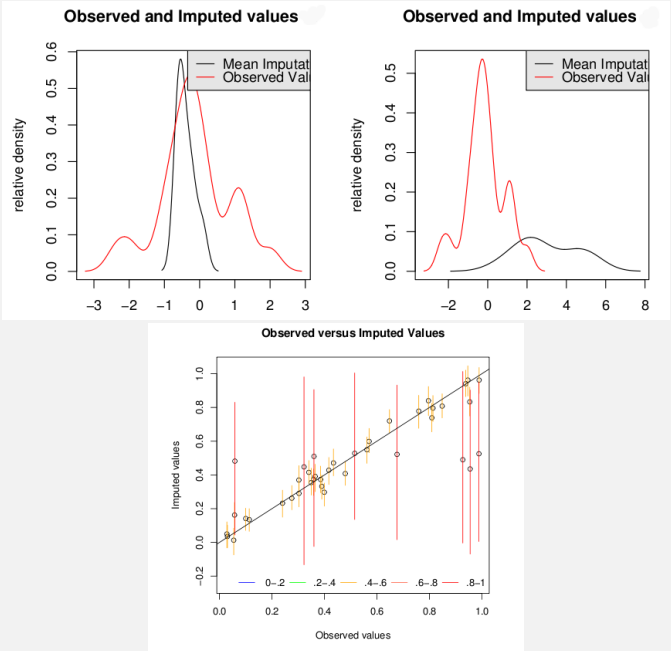
- rggobi: Interface between R and GGobi (missing data tools)
- SeqKnn: Sequential KNN imputation method
- SimHap: A comprehensive modeling framework for epidemiological outcomes and a multiple-imputation approach to haplotypic analysis of population-based data
- VIM: Visualization and Imputation of Missing Values (nearest-neighbour, hot-deck, Monte-Carlo)
- yaImpute: An R Package for k-NN Imputation

I chose Amelia II, as it conducts multiple imputation using a simple, intuitive GUI, including diagnostics and priors.

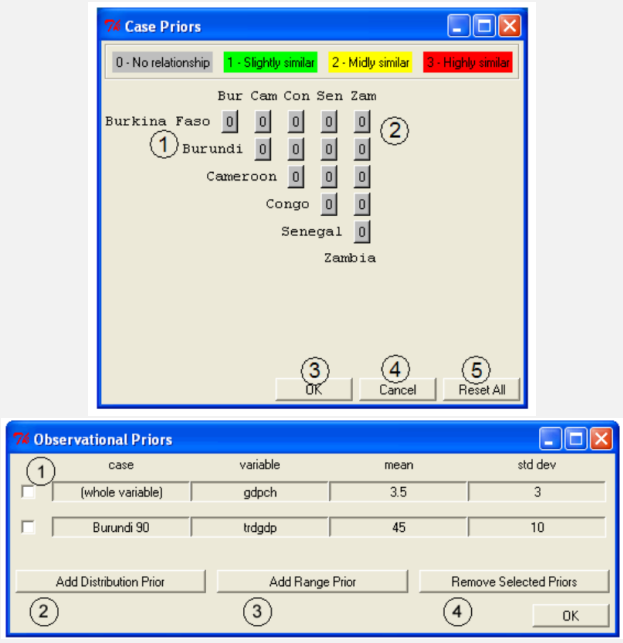
Multiple imputation using Amelia II: Simple Interface



Multiple imputation using Amelia II: Diagnostics



Multiple imputation using Amelia II: Priors



An important aspect of the data imputation method is that it allows ALL available data to be used in the imputation - even data that is not used in the final model (see Pampaka, Hutcheson and Williams, 2014)

What multiple imputation does...

1. imputation: impute m datasets using all available data
2. analysis: compute the desired analysis on all m datasets
3. combination: combine the results from m models

The imputations can be completed using the GUI. However, to save time, the R-code to achieve the analysis is shown below...

R-code for computing multiple imputations

Load the dataset 'data'

```
> tail(data)

     Course Disposition DROPout1 GCSEgrade SelfEfficacy
1369 ASTrad        4.32     <NA>       5_A         2.96
1370 ASTrad        4.32     <NA>       5_A        -0.21
1371 ASTrad        1.63     <NA>       5_A         0.10
1372 ASTrad        4.32     <NA>      6_A*         3.59
1373 ASTrad        0.50     <NA>       5_A         0.43
1374 ASTrad        2.83     <NA>      6_A*         0.54
```

The task is to impute the missing values on variable DROPout1

Define the categorical variables as numeric (this is a requirement for the imputation process).

```
> data$DROPout1.num <- as.numeric(data$DROPout1)
> data$Course.num <- as.numeric(data$Course)
> data$GCSEgrade.num <- as.numeric(data$DROPout1)
```

Run the data imputation (this command has been copied from the output of the GUI)...

```
> require(Amelia)
> imputed.datasets <- amelia(data, m=5,
+     idvars = c("Course", "DROPout1", "GCSEgrade"),
+     noms="DROPout1.num")
```

This imputes 'm' datasets and saves to 'imputed.datasets'

Each individual dataset can be accessed individually...

```
> tail(imputed.datasets$imputations[[3]])
     Course Disposition DROPout1 GCSEgrade DROPout1.num
1369 ASTrad        4.32     <NA>       5_A            2
1370 ASTrad        4.32     <NA>       5_A            1
1371 ASTrad        1.63     <NA>       5_A            1
1372 ASTrad        4.32     <NA>      6_A*            1
1373 ASTrad        0.50     <NA>       5_A            2
1374 ASTrad        2.83     <NA>      6_A*            2

> tail(imputed.datasets$imputations[[5]])
     Course Disposition DROPout1 GCSEgrade DROPout1.num
1369 ASTrad        4.32     <NA>       5_A            2
1370 ASTrad        4.32     <NA>       5_A            2
1371 ASTrad        1.63     <NA>       5_A            1
1372 ASTrad        4.32     <NA>      6_A*            1
1373 ASTrad        0.50     <NA>       5_A            2
1374 ASTrad        2.83     <NA>      6_A*            1
```

In order to run a logit model, define the imputed variable 'DROPout1.num' as categorical for each imputed dataset...

```
imputed.datasets$imputations[[1]]$DROPout1.num <-
   as.factor(imputed.datasets$imputations[[1]]$DROPout1.num)

imputed.datasets$imputations[[2]]$DROPout1.num <-
   as.factor(imputed.datasets$imputations[[2]]$DROPout1.num)

....
....

imputed.datasets$imputations[[5]]$DROPout1.num <-
    as.factor(imputed.datasets$imputations[[5]]$DROPout1.num)
```

Note: there are more elegant ways to do this - however, it is late and I am tired!

Now run the model for each imputed dataset...

```
> require(Zelig)
> Zelig.model.imp <- zelig(DROPout1.num ~ Course +
+   Disposition +  GCSEgrade + SelfEfficacy,
+   model = "logit", data = imputed.datasets$imputations)
```

and combine them into a single model...

```
> summary(Zelig.model.imp, subset = 1:5)
```

or print out individual models...

```
> print(summary(Zelig.model.imp), subset = 2:3)
```

```
> summary(Zelig.model.imp, subset = 1:5)

  Model: logit
  Number of multiply imputed data sets: 5

Coefficients:
                            Value Std. Error     t-stat      p-value
(Intercept)              1.26126222 0.27140839  4.6471010 2.653560e-04
Course[T.UoM]           -0.89892853 0.24953293 -3.6024445 4.890800e-03
Disposition             -0.08012430 0.04807018 -1.6668194 1.307939e-01
GCSEgrade[T.2_higherC] -0.43219708 0.31568835 -1.3690625 1.746078e-01
GCSEgrade[T.3_intB]    -0.74082693 0.25588026 -2.8952094 8.423871e-03
GCSEgrade[T.4_higherB] -1.06042772 0.25306183 -4.1903899 2.047710e-04
GCSEgrade[T.5_A]       -1.65856513 0.32481516 -5.1061814 1.620023e-04
GCSEgrade[T.6_A*]      -2.89299130 0.50020612 -5.7835984 9.973941e-06
SelfEfficacy            -0.08539167 0.10780845 -0.7920685 4.544826e-01

For combined results from datasets i to j, use summary(x, subset = i:j).
For separate results, use print(summary(x), subset = i:j).
```


Modelling Dropout using retrieved data

The results from Pampaka, Hutcheson and Williams, 2014...

	n=495 Original data	n=1374 Retrieved Data	n=1374 Imputed Data
CourseUoM	-1.15 (<.001)	-1.29 (<.001)	-0.87 (<.001)
GCSE_HC	-0.44 (0.44)	-0.26 (0.38)	-0.36 (0.29)
GCSE_intB	-0.46 (0.16)	-0.88 (<.001)	-0.64 (0.007)
GCSE_HB	-0.67 (0.05)	-1.02 (<.001)	-0.95 (<.001)
GCSE_A	-1.85 (<.001)	-2.25 (<.001)	-1.55 (<.001)
GCSE_A*	-4.90 (<.001)	-2.74 (<.001)	-3.83 (<.001)
Disposition	-0.09 (0.06)	-0.13 (<.001)	-0.08 (0.06)
SelfEfficacy	-0.07 (0.49)	-0.18 (<.01)	-0.06 (0.48)

Conclusions

The model using the imputed data is 'closer' to the model using the retrieved data than the model using the original data.

Even with very poor data (nearly two-thirds missing data on a binary variable; imputation produces a better model.

The R imputation software allows a detailed, considered treatment of missing data using readily available, non-technical interfaces.