MISSING DATA

Research Methods in Psychology I & II ▪ Department of Psychology ▪ Colorado State University

BY THE END OF THIS UNIT YOU WILL:

1. Know about the types of missing data.
2. Understand missing data mechanisms.
3. Be aware of various methods used to handle missing data and the assumptions that go along with each method.
4. Learn how to use multiple imputation in R with the mice package.

What is missing data?

Data are missing when observations that should be part of your dataset are not available. Missing data are a common challenge experienced by analysts. The proper handling of missing helps to ensure that we do not unnecessarily lose power and our resultant models are not biased. In this unit, you will learn about techniques for handling missing data.

ID	Y	X1	X2	X3
1	32	0	6	5
2	25	1	5	3
3	40	1	7	6
4	?	?	?	?
5	5	1	4	2
6	27	0	?	7
7	35	0	3	?
8	30	?	1	3
9	7	1	2	7
10	14	1	7	1

Set Up Notebook

To begin, please set up a new notebook called MissingDataNotebook, and save it in your MyClassActivities folder. **We need to install a new package called mice: install.packages("mice")**

Load libraries

```
library(tidyverse)
library(mice)
library(GGally
library(olsrr)
```

Import the data

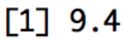
```
jobhypo <- read_csv("jobhypo.csv")
boys <- read_csv("boys.csv")
```

Start with a Simple Example

Imagine that a researcher is interested in employee job performance (y). The job performance scores for 5 randomly selected employees are shown below. Because all observations are observed, we can calculate the mean of *y* without issue.

Mean with complete data

```
y <- c(9, 13, 10, 8, 7)
mean(y)
```

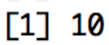


```
[1] 9.4
```

Now, imagine that one of the cases is missing on *y*. If we want to calculate the mean of *y*, we have to make a choice. Often that choice is to delete the missing case.

Mean with incomplete data

```
ymiss <- c(9, 13, 10, 8, NA)
mean(ymiss, na.rm=TRUE)
```

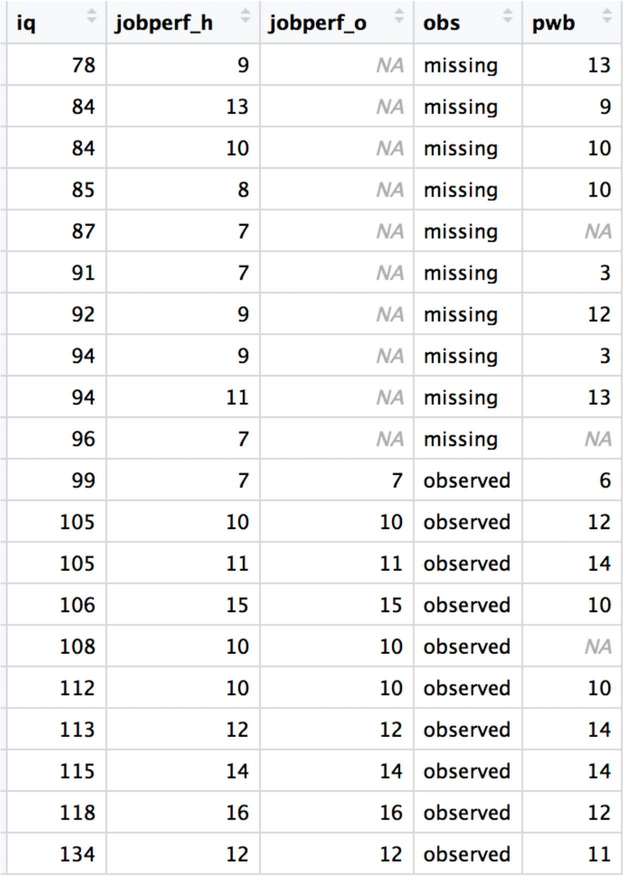


```
[1] 10
```

The mean(*y*) ≠ mean(*ymiss*). In the presence of missing data – does our estimated mean for *ymiss* generalize to the study population?

A Slightly More Complex Example

Imagine that 20 job candidates are given an IQ test (iq) and an instrument to measure psychological well-being (pwb) during a job interview. For a few of the candidates, the pwb measure was randomly lost by the company after the candidate completed it and was never recorded. The company then hires the candidates with an IQ score above the median. Six months later the supervisor rates each employee's job performance. These data are in the jobhypo dataset that we read in earlier. In the file, jobperf_o is the observed job performance for people who here hired, and jobperf_h is the hypothetical scores that we would have gotten if job performance would have been measured for all of the people.



iq	jobperf_h	jobperf_o	obs	pwb
78	9	NA	missing	13
84	13	NA	missing	9
84	10	NA	missing	10
85	8	NA	missing	10
87	7	NA	missing	NA
91	7	NA	missing	3
92	9	NA	missing	12
94	9	NA	missing	3
94	11	NA	missing	13
96	7	NA	missing	NA
99	7	7	observed	6
105	10	10	observed	12
105	11	11	observed	14
106	15	15	observed	10
108	10	10	observed	NA
112	10	10	observed	10
113	12	12	observed	14
115	14	14	observed	14
118	16	16	observed	12
134	12	12	observed	11

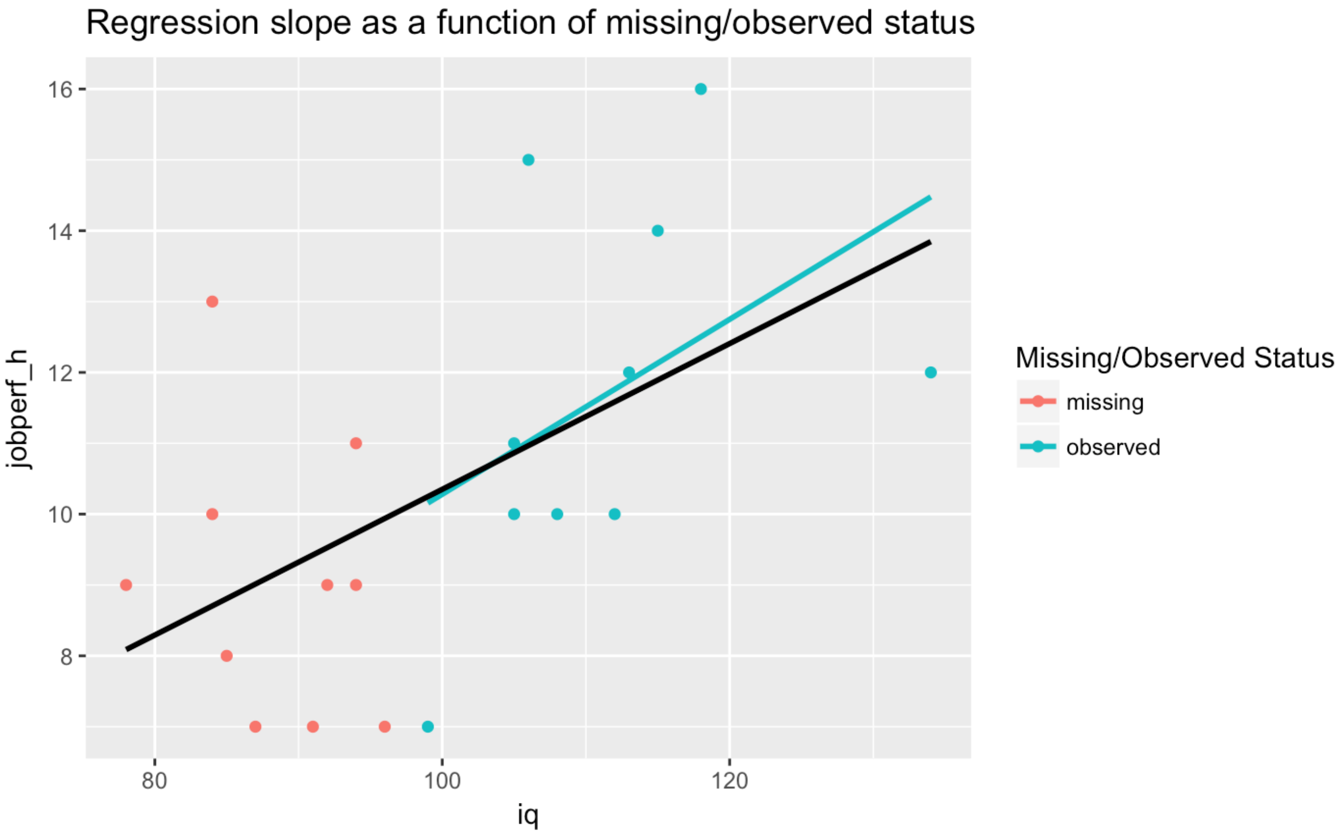
Example from: Enders, C.K. (2010). Applied missing data analysis. New York: Guilford Press.

A Plot of the jobhypo Data

Let's take a look at what the plot would like if we had all of the data vs. just the observed data. When the best fit line is fit to all of the data (black line), the slope is less steep than it is when the best fit line is fit to just the observed data. Therefore, the best fit line that we would see in a real situation wouldn't be quite right. That is, the slope of the line for the observed cases is different from what it would have been had we received all of the data.

Plot jobhypo data

```
ggplot(data = jobhypo, aes(x = iq, y = jobperf_h)) +
  geom_point(aes(color = factor(obs))) +
  geom_smooth(data = jobhypo %>% filter(jobhypo$obs == "observed"),
                        method = "lm", se = FALSE, aes(color = factor(obs))) +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  labs(title = "Regression slope as a function of missing/observed status", color = "Missing/Observed Status")
```

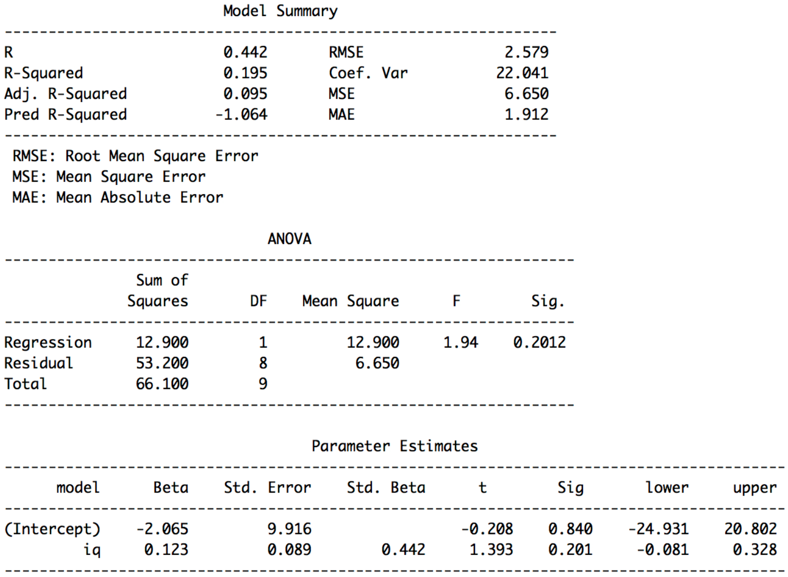


How Do the Regression Models for the jobhypo Data Differ?

Let’s fit a linear regression model to just the observed data, and then to ALL of the data to determine how our models would differ if we had access to all of the data.

fit linear regression model to observed data

```
observed <- lm(jobperf_o ~ iq, data=jobhypo)
ols_regress(observed)
```

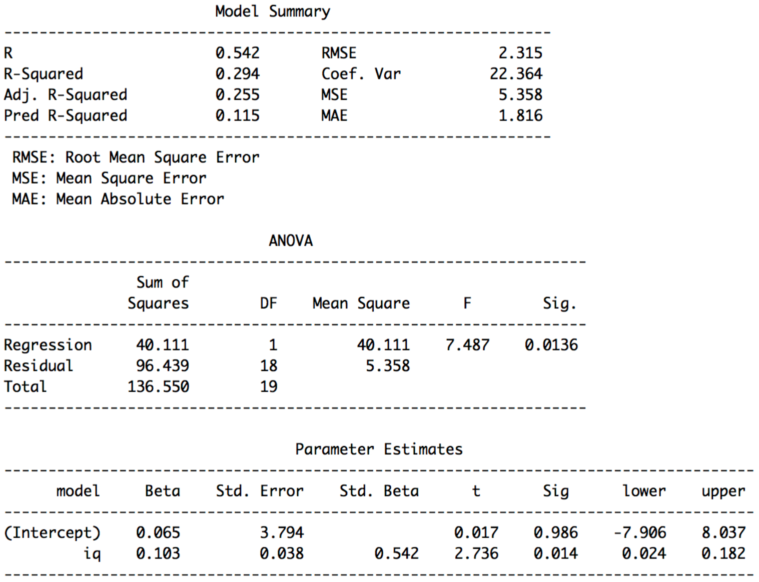


```
                        Model Summary
---------------------------------------------------------------
R                       0.442       RMSE                 2.579
R-Squared               0.195       Coef. Var           22.041
Adj. R-Squared          0.095       MSE                  6.650
Pred R-Squared         -1.064       MAE                  1.912
---------------------------------------------------------------
 RMSE: Root Mean Square Error
 MSE: Mean Square Error
 MAE: Mean Absolute Error

                              ANOVA
-----------------------------------------------------------------
               Sum of
              Squares        DF    Mean Square      F        Sig.
-----------------------------------------------------------------
Regression     12.900         1         12.900    1.94    0.2012
Residual       53.200         8          6.650
Total          66.100         9
-----------------------------------------------------------------

                                  Parameter Estimates
---------------------------------------------------------------------------------------
      model      Beta    Std. Error    Std. Beta       t        Sig       lower     upper
---------------------------------------------------------------------------------------
(Intercept)    -2.065         9.916                 -0.208    0.840    -24.931    20.802
         iq     0.123         0.089        0.442     1.393    0.201     -0.081     0.328
---------------------------------------------------------------------------------------
```

Because of the reduction in power in the observed (complete cases analysis), we fail to find a significant relationship between IQ and performance.

fit linear regression model to full data

```
full <- lm(jobperf_h ~ iq, data=jobhypo)
ols_regress(full)
```



```
                        Model Summary
--------------------------------------------------------------
R                       0.542       RMSE                2.315
R-Squared               0.294       Coef. Var          22.364
Adj. R-Squared          0.255       MSE                 5.358
Pred R-Squared          0.115       MAE                 1.816
--------------------------------------------------------------
 RMSE: Root Mean Square Error
 MSE: Mean Square Error
 MAE: Mean Absolute Error

                              ANOVA
-------------------------------------------------------------------
               Sum of
              Squares        DF    Mean Square       F         Sig.
-------------------------------------------------------------------
Regression     40.111         1         40.111    7.487    0.0136
Residual       96.439        18          5.358
Total         136.550        19
-------------------------------------------------------------------

                                 Parameter Estimates
------------------------------------------------------------------------------------
      model     Beta    Std. Error    Std. Beta      t        Sig      lower    upper
------------------------------------------------------------------------------------
(Intercept)    0.065         3.794                 0.017    0.986    -7.906    8.037
         iq    0.103         0.038        0.542    2.736    0.014     0.024    0.182
------------------------------------------------------------------------------------
```

Questions to Consider when Handling Missing Data

Given our missing data scenario…

1. Do our model estimates generalize to the population of interest?
2. Has missing data affected our power to detect an effect?
3. If the rates of missingness differs across predictors:

    A. Can we compare models?

    B. Are differences in the coefficients due to changes in the model or changes in the sample?

4. Are we making the best use of our data?

Most studies produce at least some missing data. At the study planning and data collection phases, putting a plan into place to prevent missing data is key. When missing data does occur, proper handling of it is necessary. Ignoring or improperly handling missing data can result in biased estimates (different from what they should be if our model were proper), incorrect standard errors, and incorrect inferences.

van Buuren, S. (2012). Flexible Imputation of Missing Data. Boca Raton, FL: Chapman & Hall.

Missing Data Mechanisms

The missing data mechanism is the cause of the missing data. Mechanisms describe how the missing value is related to other observed and unobserved variables, and the variable itself.

The missing data mechanism has implications for how to handle missing data. Missing data mechanisms are grouped into 3 categories: **Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR).**

Missing Completely at Random

When missing data are MCAR, the cases with missing values are a random sample of the original sample. The cases with missing values do not systematically differ from those without missing values. In addition, missingness is not related to any factor, known or unknown, in the study. We know for sure that MCAR fails if the likelihood of being observed vs. missing can be predicted. You can get a sense for whether data are MCAR by determining if there are differences on observed variables between those missing on a certain variable and those who are observed. For example, average IQ is clearly lower for people who are missing on job performance; therefore, the missingness on job performance is not MCAR.

In thinking about the IQ and Job Performance example, the missing data mechanism for pwb is MCAR because the data are missing totally haphazardly — that is, a few of the assessments were simply randomly lost. We can calculate average IQ for those who are missing on pwb and those who are not missing — MCAR would be plausible if there was no mean difference in IQ between the two groups. However, the assumption is that missing is unrelated to ALL observed and unobserved variables — and the latter is untestable.

Another example:

- A physician is studying the effect of a new treatment on blood pressure. One day, the device used to measure blood pressure went haywire and wouldn't record responses. The blood pressure data from study participants who were assessed on that one day were lost.

Missing At Random

Also called Accessible Mechanism. When data are MAR, the cases with missing values do systematically differ from those without missing values – but we observe the ways in which they differ. Once all of the known causes of missingness are taken into account, any residual missingness can be considered MCAR. The missing data technique that we will consider today, Multiple Imputation, assumes the data are MAR. You can't prove data are MAR, but including a relatively rich set of predictors in the missing data model (auxiliary variables), can make the MAR assumption more plausible.

In thinking about the IQ and Job Performance example, the data on job performance were missing at random. We know that job performance was missing because the individual scored in the lower half of the IQ distribution.

Another example:

- A questionnaire about health behaviors is given to children at a school. Some students do not finish the questionnaire, leaving many questions blank. However, it is determined that the children with poor reading skills often do not make it to the end. Including reading achievement (which is available via official school records) in the missing data model will be important.

Missing Not At Random

Also called Inaccessible Mechanism. When data are MNAR, the cases with missing values do systematically differ from those without missing values – but we do not observe the important ways in which they differ. In this case, missingness depends on the would be missing value. The mechanism for the missing data is inaccessible because we know data are missing but we do not know why.

This type of missing requires special techniques (e.g., pattern mixture models, selection models) – and often these issues can't be solved. Please note that MAR becomes MNAR if you fail to include the observed mechanism for missingness in your model.

In thinking about our IQ and Job Performance example, missing data would be MNAR if everyone was hired, but low performers didn't show up for their performance assessment.

Another example:

- A researcher is studying substance use among college students. Students are invited to report to the lab to complete a time-line follow-back assessment of their drinking behaviors. Heavy drinkers are out drinking during their scheduled appointment and don't show up to complete the assessment.

Undesirable Techniques for Handling Missing Data: Listwise Deletion

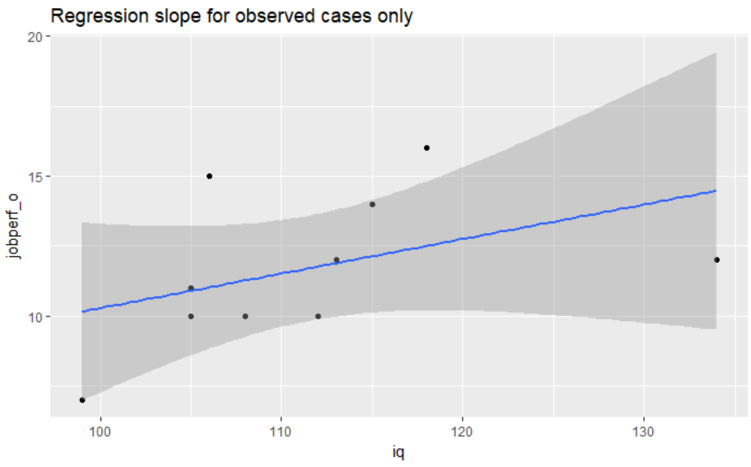
Eventually, we will learn about proper techniques for handling missing data. But first, let's consider some undesirable techniques.

Listwise deletion is the most commonly employed techniques.   With listwise deletion (also called complete cases analysis) all cases with any missing data on the predictors or the outcome of your model are deleted. This is the default in most statistical software. This technique assumes data are MCAR. If MCAR is not the case, complete cases analysis can lead to biased estimates, decreased power, and compromised generalizability. Besides this issue, complete cases analysis is problematic if you are fitting a series of models because each model will have a different N. Even if data are MCAR, complete cases analysis negatively impacts power.

There is a similar method called pairwise deletion that deletes cases in a pairwise fashion. A correlation matrix is an example of this. For each correlation in a matrix, cases with missing data on either of the two variables making up the variable are discarded.

Complete cases example

```
ggplot(data = jobhypo %>% filter(obs == "observed"), aes(x = iq, y = jobperf_o)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Regression slope for observed cases only")
```

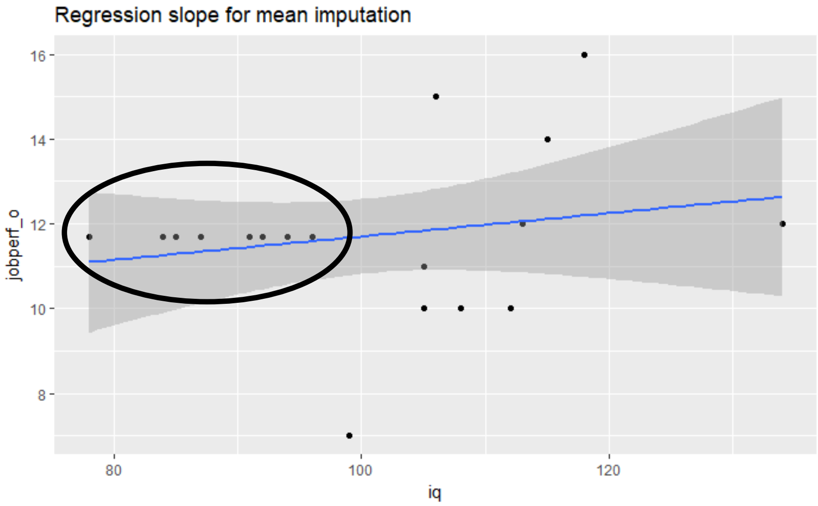


Undesirable Techniques for Handling Missing Data: Mean Imputation

This technique is also referred to as arithmetic mean imputation, unconditional mean imputation, or mean substitution. Here, the arithmetic mean of the available cases for a variable (y) is substituted for all cases with a missing value on y. Mean imputation will reduce the variability of the data and can then artificially reduce the standard deviation and variance of the variable itself. Mean imputation will also affect the covariance (and correlation) of the mean substituted variable with other variables. This technique can bias the estimates of a model even when MCAR is met. **This is probably the worst technique you can apply to deal with missing data.**

Mean imputation example

```
imp_m <- mice(impd, method = "mean", m=1, maxit = 1)

imp_md <- mice::complete(imp_m, "long")

ggplot(imp_md, aes(x = iq, y = jobperf_o)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Regression slope for mean imputation")
```

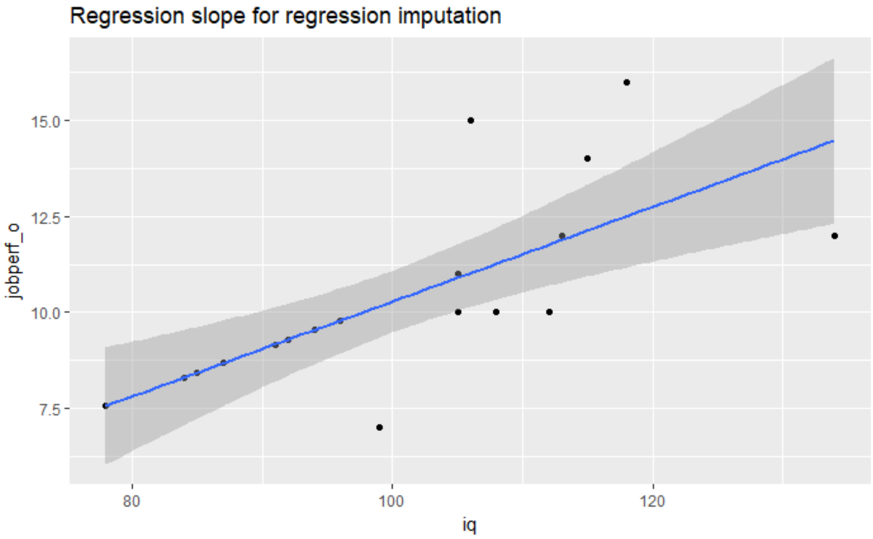


Undesirable Techniques for Handling Missing Data: Regression Imputation

This technique is also referred to as conditional mean imputation. Here, a regression model is specified using the complete cases. The regression estimates are then used to get the predicted score for cases with missing values. While this approach is better than mean imputation, it can still produce biased estimates. Notice that for the missing cases, the correlation between IQ and job performance is 1 — an unrealistic result. Therefore, regression imputed data will result in inflated correlations between variables and attenuated variances and covariances.

Regression imputation example

```
imp_r <- mice(impd, method = "norm.predict", m = 1, maxit = 1, seed = 1)

imp_rd <- mice::complete(imp_r, "long")

ggplot(imp_rd, aes(x = iq, y = jobperf_o)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Regression slope for regression imputation")
```

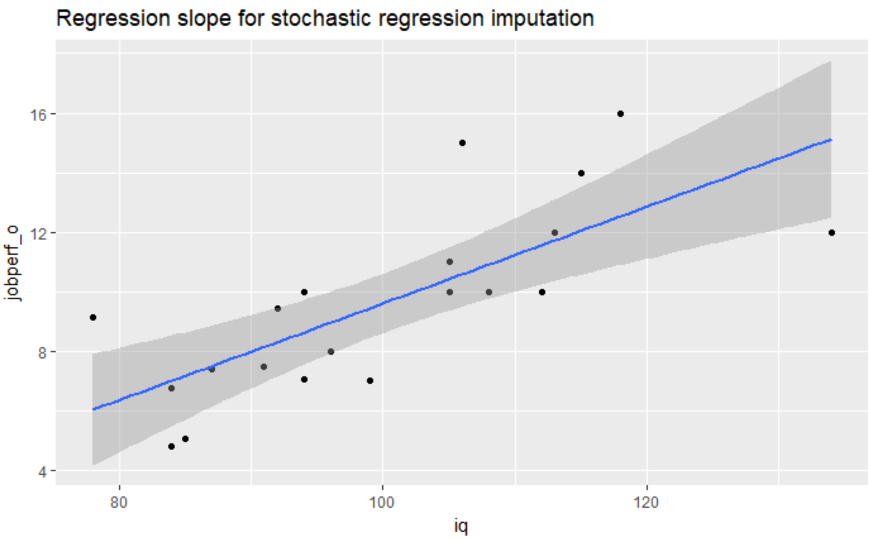


Undesirable Techniques for Missing Data: Stochastic Regression Imputation

Stochastic regression imputation attempts to improve on regression imputation by adding a normally distributed residual term to each predicted score. This restores lost variability. Under the assumption of MAR, this method can actually produce unbiased estimates. Of all the methods discussed so far, this is the best. However, when we proceed to analyze the imputed data, the "filled-in" cases will be treated as real data and the uncertainty associated with the predicted value because it was missing will not be properly accounted for. As a result, the standard errors from a regression model in which jop performance is regressed on IQ will be too small. As you will soon see, building on this approach by performing the stochastic regression imputation MULTIPLE times is the basis for an excellent missing data handling technique called multiple imputation.

Stochastic regression imputation example

```
imp_sr <- mice(impd, method = "norm.nob", m = 1, maxit = 10, seed = 1)

imp_srd <- mice::complete(imp_sr, "long")

ggplot(imp_srd, aes(x = iq, y = jobperf_o)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Regression slope for stochastic regression imputation")
```



What Characteristics Should a Proper Technique for Missing Data Possess?

- **Account for the process that created the missing data.** We need a method that allows us to incorporate missing data mechanisms — e.g, other variables that can do a good job of telling us about what the missing values might have been if they had been observed and variables related to the missing data mechanism.
- **Preserve the relations in the data.** We need a method that can assess the multivariate distribution of the observed data and preserve those elements.
- **Preserve the uncertainty about these relations.** When we impute the data, we need a method that can adequately represent our uncertainty about the true values of the missing data.

There are three commonly employed and proper techniques for handling missing data:

- **Multiple Imputation.** This is what we'll cover today using the mice package in R.
- **Full Information Maximum Likelihood.** This is a technique commonly used in structural equation models.
- **Inverse Probability of Response Weighting.** To carry this out, the analyst creates a model to predict non-response given observed covariates, and then weights respondents by their inverse probability of response. This is a technique that is often used to account for dropout in studies of treatment effects.

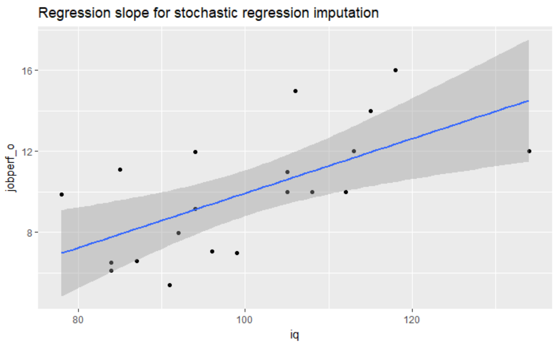
Multiple Imputation (MI)

Multiple imputation builds on the idea of stochastic regression imputation – but fills in each missing value multiple times (e.g., creates 20 complete datasets). When employing MI, the analyst builds a model for missing data. All of the variables in the analysis model, plus auxiliary variables that predict or are correlated with missingness and missing variables should included in the imputation model.

Once the multiple imputations are created. Analyses are run on each complete dataset and the results are combined using Rubin's Rules (Rubin, 1987). In this way, the total variance in the final regression estimates is a function of within-imputation variance and between-imputation variance. Thus, the uncertainty in the imputed values is taken into account.

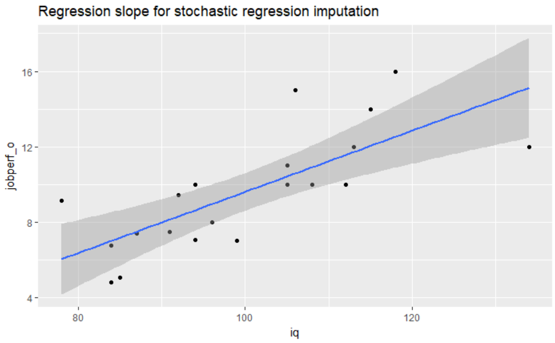
Rubin, D.B. (1987). Multiple Imputation for Nonresponse in Surveys. New York" John Wiley & Sons.

Imputation 1

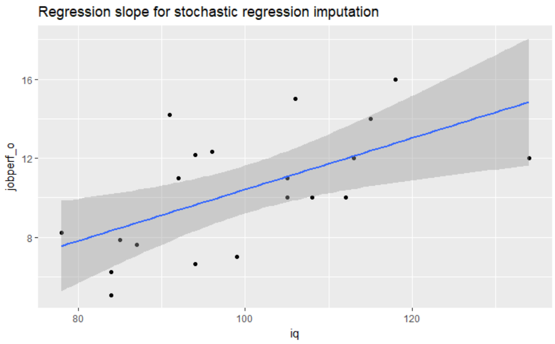


Within imputation variability is captured by the stochastic process of taking the predicted score and adding a residual.

Imputation 2



Imputation 3



The between imputation variability is captured because the imputed values are a bit different in each imputation.

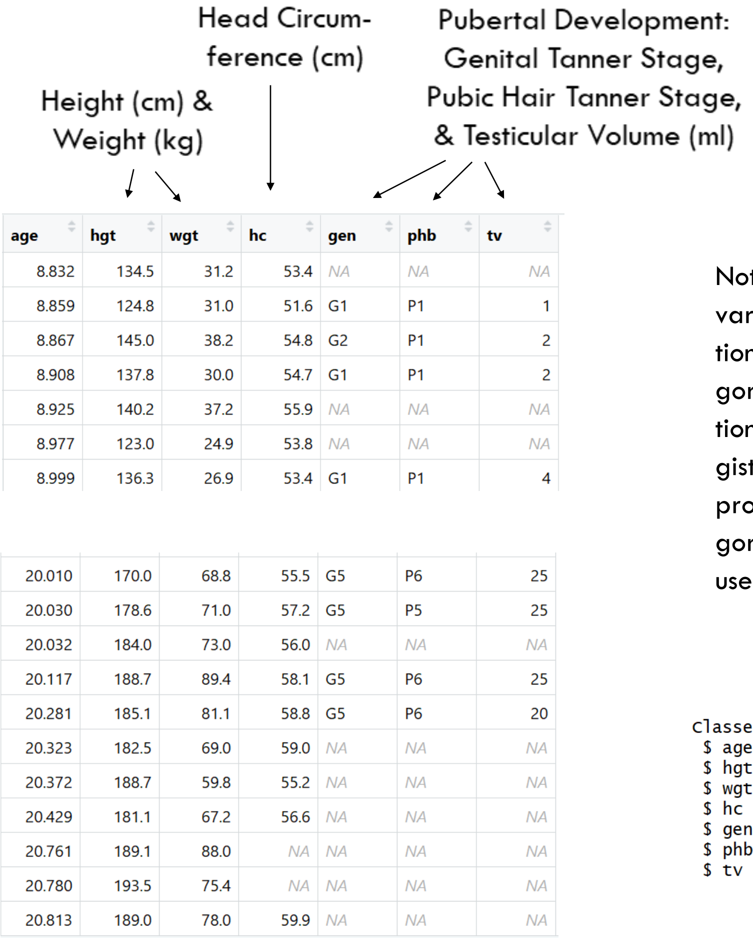
An Example: Dutch Growth Study

The Dutch Growth Study is a cross-sectional, nationwide study of physical growth among children and adolescents 0 to 21 years of age. We will consider boys between the ages of 8 and 21 (N=424). There were 180 boys (42%) with missing data on pubertal development. This missingness is strongly related to age (20% missing at age 9-11, 60% missing at ages 17-20). This example is from Stef van Buuren's mice vignette.

Prepare the boys data

```
boys1 <- boys %>%
  filter(age >= 8.0 & age <= 21.0) %>%
  select(-bmi, -reg) %>%
  mutate(gen = ordered(gen, levels = c("G1", "G2", "G3", "G4", "G5")),
      phb = ordered(phb, levels = c("P1", "P2", "P3", "P4", "P5", "P6")))

str(boys1)
```

In preparation for the imputation, we will discard boys outside of our designated age range. We also will discard two variables that we won't use in our imputation. We want to arrive at a subset of our data that has just the variables that will be used for the imputation.



Height (cm) & Weight (kg)

Head Circumference (cm)

Pubertal Development: Genital Tanner Stage, Pubic Hair Tanner Stage, & Testicular Volume (ml)

age	hgt	wgt	hc	gen	phb	tv
8.832	134.5	31.2	53.4	NA	NA	NA
8.859	124.8	31.0	51.6	G1	P1	1
8.867	145.0	38.2	54.8	G2	P1	2
8.908	137.8	30.0	54.7	G1	P1	2
8.925	140.2	37.2	55.9	NA	NA	NA
8.977	123.0	24.9	53.8	NA	NA	NA
8.999	136.3	26.9	53.4	G1	P1	4
20.010	170.0	68.8	55.5	G5	P6	25
20.030	178.6	71.0	57.2	G5	P5	25
20.032	184.0	73.0	56.0	NA	NA	NA
20.117	188.7	89.4	58.1	G5	P6	25
20.281	185.1	81.1	58.8	G5	P6	20
20.323	182.5	69.0	59.0	NA	NA	NA
20.372	188.7	59.8	55.2	NA	NA	NA
20.429	181.1	67.2	56.6	NA	NA	NA
20.761	189.1	88.0	NA	NA	NA	NA
20.780	193.5	75.4	NA	NA	NA	NA
20.813	189.0	78.0	59.9	NA	NA	NA

Notice that in the code above we also specify that two of our variables are ordered categorical (gen and phb). In the imputation phase, we want MICE to know that these are ordered categorical variables. MICE will automatically pick the right link function for the regression models (e.g., identity for continuous, logistic regression for categorical, etc.), so we can streamline the process if we specify the type of variable. For unordered categorical variables, you can use the factor function that we have used many times in the course.

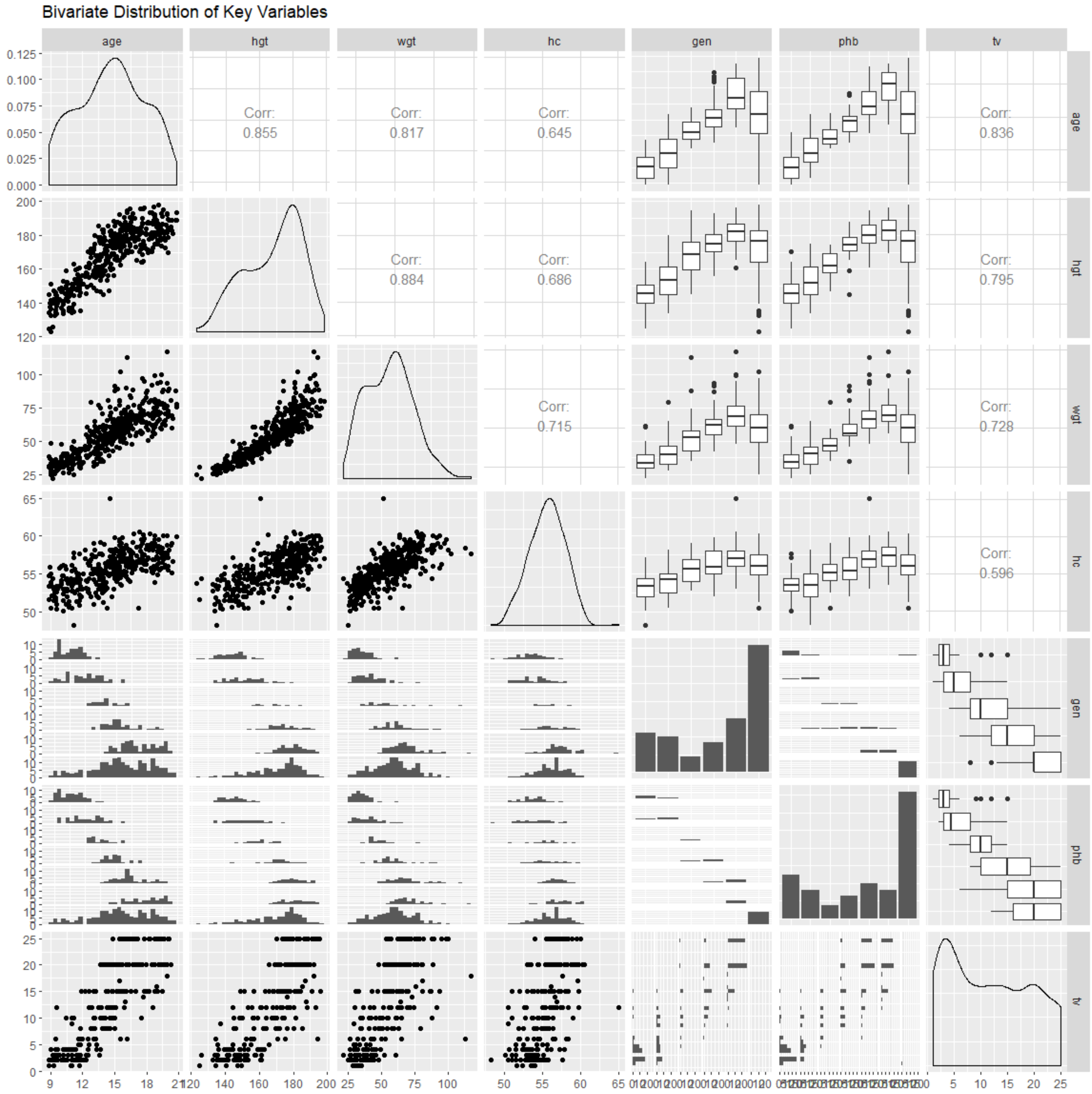
```
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':      424 obs. of  7 variables:
 $ age: num  8.83 8.86 8.87 8.91 8.93 ...
 $ hgt: num  134 125 145 138 140 ...
 $ wgt: num  31.2 31 38.2 30 37.2 24.9 26.9 48.2 29.4 30 ...
 $ hc : num  53.4 51.6 54.8 54.7 55.9 53.8 53.4 50.5 53.9 53.6 ...
 $ gen: Ord.factor w/ 5 levels "G1"<"G2"<"G3"<..: NA 1 2 1 NA NA 1 2 1 NA ...
 $ phb: Ord.factor w/ 6 levels "P1"<"P2"<"P3"<..: NA 1 1 1 NA NA 1 1 1 NA ...
 $ tv : int  NA 1 2 2 NA NA 4 2 2 NA ...
```

Let’s Take a Look at the Scatterplot Matrix

Scatterplot matrix

```
ggpairs(boys1, title = "Bivariate Distribution of Key Variables")
```

Add: ig.height = 12, fig.width = 12, to the code chunk improve sizing.



Examine the Missing Data Patterns

We can use the md.pattern function in MICE to take a look at the various missing data patterns.

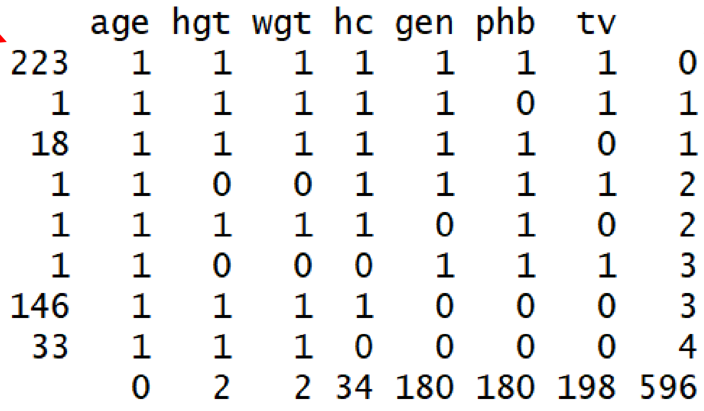
Missing data patterns

```
md.pattern(boys1)
```

1= observed, 0 = not observed

# of cases for each type

# of missing variables for each type



```
    age hgt wgt hc gen phb  tv    
223   1   1   1  1   1   1   1   0
  1   1   1   1  1   1   0   1   1
 18   1   1   1  1   1   1   0   1
  1   1   0   0  1   1   1   1   2
  1   1   1   1  1   0   1   0   2
  1   1   0   0  0   1   1   1   3
146   1   1   1  1   0   0   0   3
 33   1   1   1  0   0   0   0   4
      0   2   2 34 180 180 198 596
```

# of cases missing for each variable, with the last entry being the total number of missing data points

Multiple Imputation in 5 Steps

1. Build the imputation model
2. Impute the data (create the multiply imputed datasets)
3. Evaluate the imputations (determine that the imputations are tenable)
4. Estimate the model of interest within each imputed dataset
5. Combine the results from each imputed dataset to arrive at a final set of estimates

Step 1: Build the Imputation Model

- Include all variables (predictors and outcomes) that will be in the analysis model.
- The missing data model should be at least as general as the analysis model. So, if you are proposing an interaction between two variables, include the interaction term as a variable in your imputation model. If your moderator is categorical, you can also impute separately for each group (i.e., differential effects by sex are proposed, so create a set of imputations for males and a set of imputations for females — then combine the sets for analysis).
- Include auxiliary variables that aren't going to be in the analysis model but are either predictive of non-response or are substantially correlated with the items with missing data.
- Use a large number of imputations (initial advice was 5-10, but more (e.g., 40) may increase power at very little cost). More missing data requires more imputations.

Step 2: Impute the data

There are several methods for creating the imputations. The two most common are:

- Joint modeling of all variables under a multivariate normal distribution. A Markov Chain Monte Carlo (MCMC) algorithm is used to obtain imputed values from the estimated multivariate normal distribution of the observed data.
- Multiple Imputation by Chained Equations (MICE). With this method, a model for each variable is fit, conditional on all other variables in the missing data model. MICE employs a Gibbs Sampling Algorithm for fitting the model and imputing variables. One great feature of a MICE approach is that the model for each variable is easily fit using the proper distribution (e.g., continuous, categorical). There is a nice package called mice in R that employs this technique. There is also an add on to the package for multilevel data (https://cran.r-project.org/web/packages/micemd/micemd.pdf). This is the technique that we will learn about in this unit.

mice function and arguments

Get help

```
?mice
```

```
mice(data, m = 5, method = vector("character", length = ncol(data)),  predictorMatrix = (1 - diag(1, ncol(data))),
visitSequence = (1:ncol(data))[apply(is.na(data), 2, any)],  form = vector("character", length = ncol(data)),
post = vector("character", length = ncol(data)), defaultMethod = c("pmm",   "logreg", "polyreg", "polr"), maxit = 5, diagnostics = TRUE, printFlag = TRUE, seed = NA, imputationMethod = NULL,  defaultImputationMethod = NULL, data.init =
NULL, ...)
```

defaultMethod
A vector of three strings containing the default imputation methods for numerical columns, factor columns with 2 levels, and columns with (unordered or ordered) factors with more than two levels, respectively. If nothing is specified, the following defaults will be used: pmm, predictive mean matching (numeric data) logreg, logistic regression imputation (binary data, factor with 2 levels) polyreg, polytomous regression imputation for unordered categorical data (factor >= 2 levels) polr, proportional odds model for (ordered, >= 2 levels)

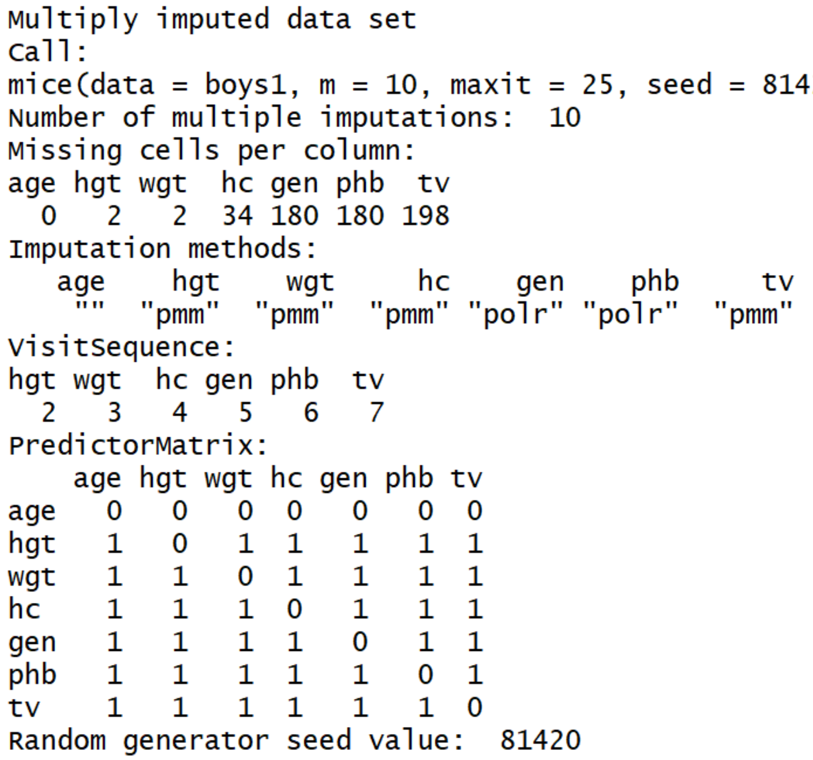
Very generally, this is how mice works.   An imputation model is specified separately for each variable, in which all of the other variables serve as predictors. At each stage of the algorithm, an imputation is generated for the missing variable, then this imputed value is used in the imputation of the next variable. This process repeats, imputing missing values using a Gibbs sampling procedure until the process reaches convergence. Separate chains are used to generate the multiple imputations, once all converge the process is complete.

Use mice to Create Multiple Imputations

Impute data

```
fcs <- mice(boys1, seed = 81420, m = 10, mincor = 0, maxit = 25)
fcs
```

Since this is an introduction to mice, we will mostly go with the defaults. I am changing just a few things so you get a sense of how to modify. First, we will set the seed so that we get the same result each time model is executed. Second, the default number of imputations is 5, but I increase this to 10 (m = 10), in practice you can do as many as you like. Third, mincor provides the minimum correlation between a predictor and an outcome that you want to require in order for the predictor to be used in the imputation of a particular variable. The default is .10, I change this to 0 which will force all variables to be used to predict all outcomes. Last, maxit is the number of iterations for each imputation. I set this to 25, the default is 5. The mice package uses an iterative algorithm for imputing the data, we want to allow enough iterations so that the model for each predicted outcome converges. If inspection of the multiple imputation diagnostics indicates that the models are not converging by the number of iterations specified (we'll look at these plots soon), you can increase this number.



```
Multiply imputed data set
Call:
mice(data = boys1, m = 10, maxit = 25, seed = 81420, mincor = 0)
Number of multiple imputations:  10
Missing cells per column:
age hgt wgt  hc gen phb  tv
  0   2   2  34 180 180 198
Imputation methods:
   age    hgt    wgt     hc    gen    phb     tv
    ""  "pmm"  "pmm"  "pmm" "polr" "polr"  "pmm"
VisitSequence:
hgt wgt  hc gen phb  tv
  2   3   4   5   6   7
PredictorMatrix:
    age hgt wgt hc gen phb tv
age   0   0   0  0   0   0  0
hgt   1   0   1  1   1   1  1
wgt   1   1   0  1   1   1  1
hc    1   1   1  0   1   1  1
gen   1   1   1  1   0   1  1
phb   1   1   1  1   1   0  1
tv    1   1   1  1   1   1  0
Random generator seed value:  81420
```

For continuous variables, the default method for imputing the value is called predictive means matching. With this approach, the predicted value for the missing case is obtained based on the fitted regression model, and then the closest actually observed value is the value that is recorded. For the ordered categorical variables, the default method is an ordered logistic regression (polr—polytomous).

Step 3: Evaluate the Imputations

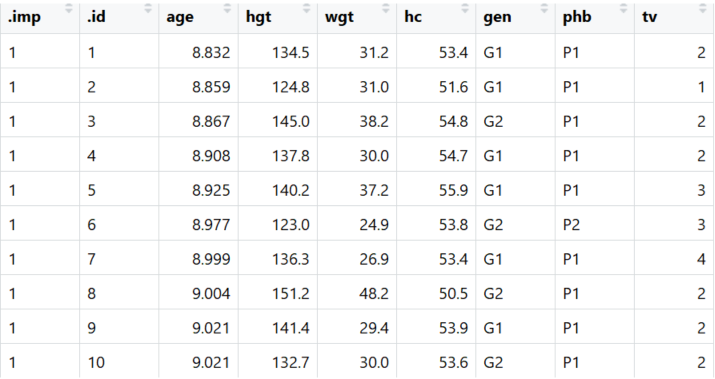
When mice executes the multiple imputation, it creates a mids object that contains a variety of different elements that will help us to evaluate the results of the imputation procedure.

Complete Dataset #1	Complete Dataset #2	Complete Dataset #3	Complete Dataset #4	Complete Dataset #5
Complete Dataset #6	Complete Dataset #7	Complete Dataset #8	Complete Dataset #9	Complete Dataset #10

Pull out the long dataset

```
stacked <- mice::complete(fcs, "long")
summary(stacked)
```

The complete argument pulls out the data from the mice object of results. Here we pull it out and create a dataset called stacked.

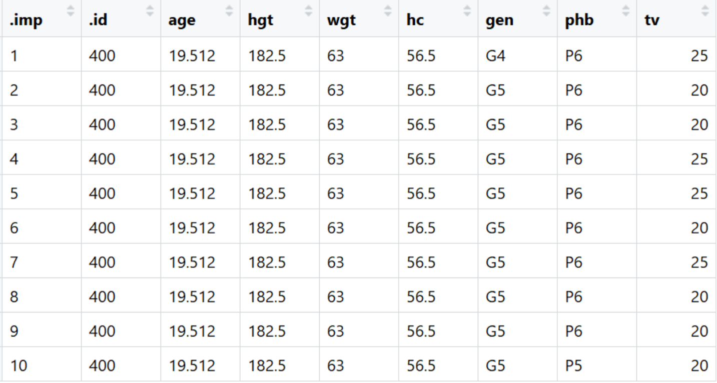


.imp	.id	age	hgt	wgt	hc	gen	phb	tv
1	1	8.832	134.5	31.2	53.4	G1	P1	2
1	2	8.859	124.8	31.0	51.6	G1	P1	1
1	3	8.867	145.0	38.2	54.8	G2	P1	2
1	4	8.908	137.8	30.0	54.7	G1	P1	2
1	5	8.925	140.2	37.2	55.9	G1	P1	3
1	6	8.977	123.0	24.9	53.8	G2	P2	3
1	7	8.999	136.3	26.9	53.4	G1	P1	4
1	8	9.004	151.2	48.2	50.5	G2	P1	2
1	9	9.021	141.4	29.4	53.9	G1	P1	2
1	10	9.021	132.7	30.0	53.6	G2	P1	2

Here's the very top of the stacked dataset. In this file are the 10 imputed datasets stacked on top of one another. A variable called .imp denotes the imputation number.

Look at one boy's data

```
id400 <- filter(stacked, .id == 400)
id400
```



.imp	.id	age	hgt	wgt	hc	gen	phb	tv
1	400	19.512	182.5	63	56.5	G4	P6	25
2	400	19.512	182.5	63	56.5	G5	P6	20
3	400	19.512	182.5	63	56.5	G5	P6	20
4	400	19.512	182.5	63	56.5	G5	P6	25
5	400	19.512	182.5	63	56.5	G5	P6	25
6	400	19.512	182.5	63	56.5	G5	P6	20
7	400	19.512	182.5	63	56.5	G5	P6	25
8	400	19.512	182.5	63	56.5	G5	P6	20
9	400	19.512	182.5	63	56.5	G5	P6	20
10	400	19.512	182.5	63	56.5	G5	P5	20

This is one boy's data. This boy was missing gen, phb, and tv. You can see how his values differ for these three variables across the imputations.

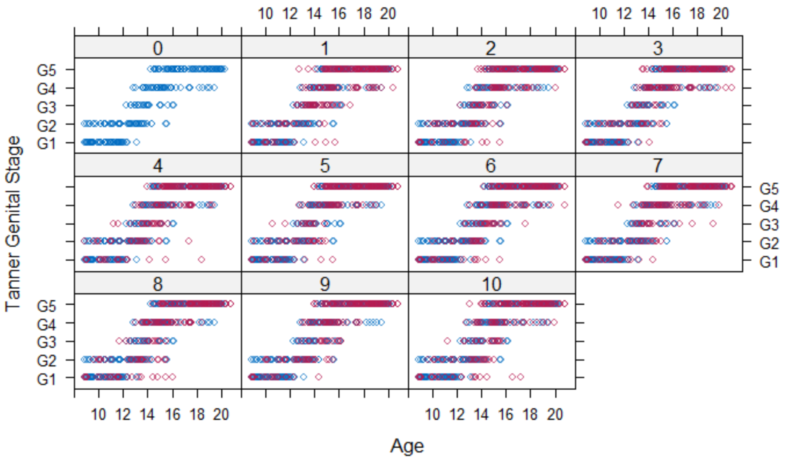
Look at Bivariate Plots

It's useful to inspect each pair of variables to make sure everything looks as expected.

Plot results for one pair of variables

```
xyplot(fcs, gen~age|.imp, xlab="Age", ylab="Tanner Genital Stage")
```

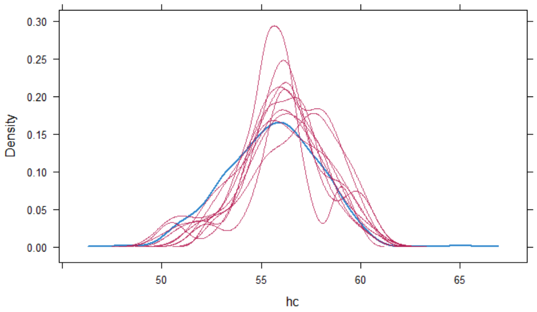
0 is the raw data, and each of the remaining panels (1-10) shows you the corresponding imputed dataset.



Look at Density Plots

Create a density plot of each variable

```
densityplot(fcs, ~hc)
```



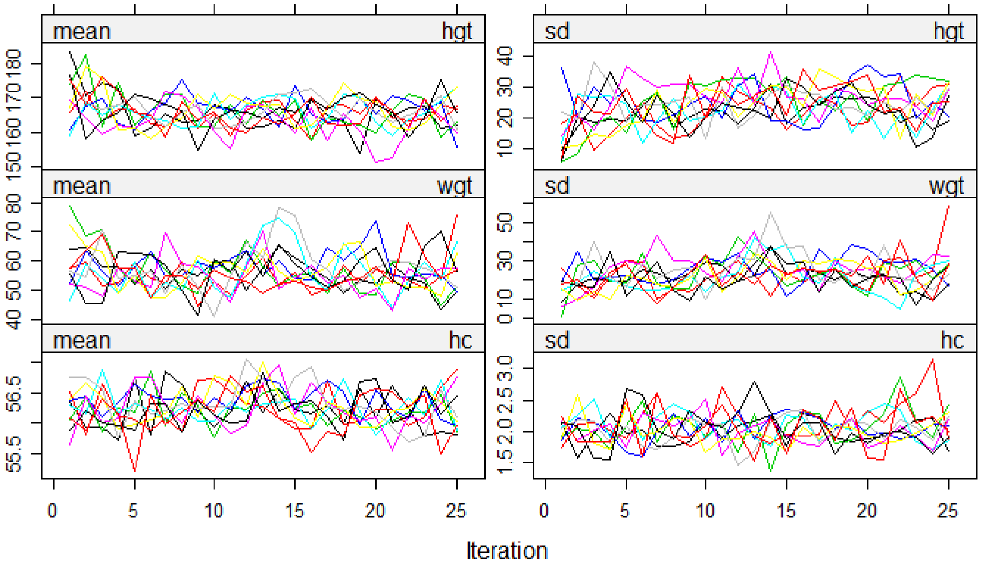
Taking a look at the density plot of each variable is also useful to make sure that the distribution across each imputed dataset (the red lines) is similar to the observed dataset (the blueline)

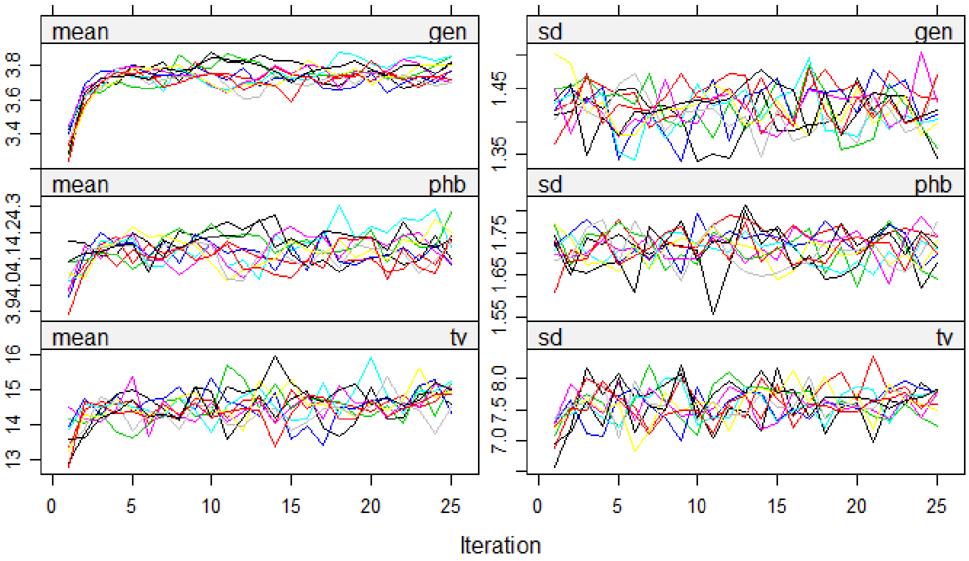
Evaluate Convergence

We need to make sure that the Gibbs sampling algorithm (the method mice uses to create the imputations) has converged. mice plots the univariate parameters (mean and sd) against the iteration number for *m* parallel imputation streams (where *m* is the number of iterations). The different streams should be freely intermingled with each other, without showing any definite trends. We want to see that by the end of the iterations, the streams have stabled out and aren't systematically snaking up and down. If they don't seem stable, then the number of iterations should be increased (e.g., maxit = 100).

Create trace plots

```
plot(fcs)
```



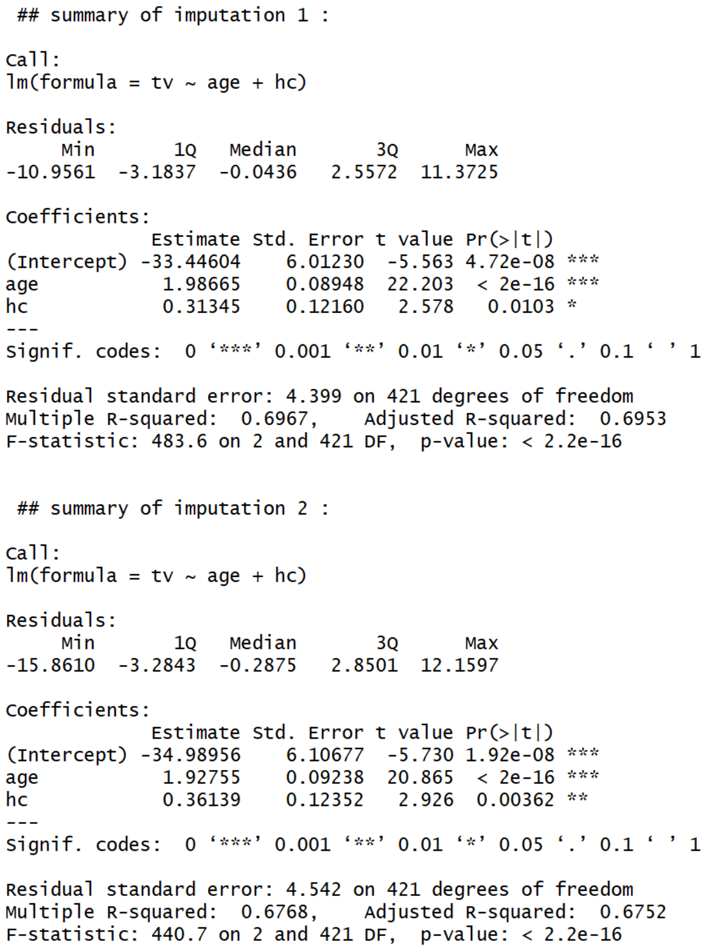


Step 4: Estimate your Model Using the Imputed Datasets

Once you feel confident with the imputations, it is time to fit the model. We will fit a linear regression model in which age and head circumference are used to predict testicular volume.

Fit a model in which tv is regressed on age and head circumference

```
fit <- with(fcs, lm(tv ~ age + hc))
summary(fit)
```



```
 ## summary of imputation 1 :

Call:
lm(formula = tv ~ age + hc)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.9561  -3.1837  -0.0436   2.5572  11.3725 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -33.44604    6.01230  -5.563 4.72e-08 ***
age           1.98665    0.08948  22.203  < 2e-16 ***
hc            0.31345    0.12160   2.578   0.0103 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.399 on 421 degrees of freedom
Multiple R-squared:  0.6967,	Adjusted R-squared:  0.6953 
F-statistic: 483.6 on 2 and 421 DF,  p-value: < 2.2e-16


 ## summary of imputation 2 :

Call:
lm(formula = tv ~ age + hc)

Residuals:
     Min       1Q   Median       3Q      Max 
-15.8610  -3.2843  -0.2875   2.8501  12.1597 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -34.98956    6.10677  -5.730 1.92e-08 ***
age           1.92755    0.09238  20.865  < 2e-16 ***
hc            0.36139    0.12352   2.926  0.00362 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.542 on 421 degrees of freedom
Multiple R-squared:  0.6768,	Adjusted R-squared:  0.6752 
F-statistic: 440.7 on 2 and 421 DF,  p-value: < 2.2e-16
```

When you execute the linear model on the mids object, you will obtain the results from each imputation. Here we see results of the first two imputations.

Step 5: Combine the Results to Obtain Final Parameter Estimates

The last step is to combine the estimates across the imputed datasets to arrive at our final model results. We use Rubin's Rules to do this.

$$\overline{Q} = \frac{1}{m}\sum_{j=1}^{m}\hat{Q}_j$$

We will get an average estimate for the intercept and each slope. It is simply the average across the imputed datasets (e.g., the average slope for hc across the 10 imputed datasets). Q = the estimate of interest (e.g., slope for hc), m = imputation #.

$$\overline{U} = \frac{1}{m}\sum_{j=1}^{m}U_j$$

This formula represents the average within-imputation variability. U = the standard error of the estimate of interest (e.g., standard error for the slope of hc).

$$B = \frac{1}{m-1}\sum_{j=1}^{m}(\hat{Q}_j - \overline{Q})^2$$

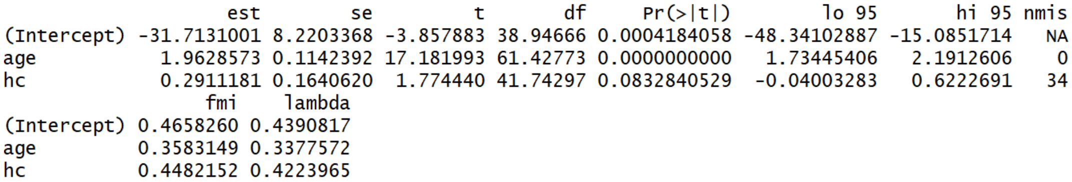
This formula represents the between imputation variability. It picks up how much the estimates vary around the average estimate. For example, how much does the slope of hc differ across the 10 imputed datasets.

$$T = \overline{U} + (1 + \frac{1}{m})B$$

Finally, the within- and between- imputation variability is combined to arrive at a final standard error that combines the uncertainty both within-imputation and between-imputations.

Pool the estimates to arrive at final results

```
est <- pool(fit)
summary(est)
```



```
                    est        se         t       df     Pr(>|t|)        lo 95       hi 95 nmis
(Intercept) -31.7131001 8.2203368 -3.857883 38.94666 0.0004184058 -48.34102887 -15.0851714   NA
age           1.9628573 0.1142392 17.181993 61.42773 0.0000000000   1.73445406   2.1912606    0
hc            0.2911181 0.1640620  1.774440 41.74297 0.0832840529  -0.04003283   0.6222691   34
                  fmi    lambda
(Intercept) 0.4658260 0.4390817
age         0.3583149 0.3377572
hc          0.4482152 0.4223965
```

The final results are interpreted in the same way as usual. For example, the slope for hc is .29. This indicates that holding constant age, each 1 unit increase in head circumference is associated with a .29 unit increase in testicular volume. The estimate divided by the standard error gives *t**. The degrees of freedom are not calculated in the usual way. These df take into account the number of imputations and the proportion of missing information. They will range between the number of imputations (10 in our example) and infinity. If the degrees of freedom are close to the number of imputations, then increasing the number of imputations is advised. The *t** and *df* are used to calculate the *p-value* in the usual way. Here, we see that the effect of head circumference is not significantly different from zero, thus, the 95% CI (-.04, .62) contains 0. The column nmis presents the number of cases with missing data on that particular predictor. The column named fmi presents the fraction of missing information in calculating the estimate, and the column named lambda is the proportion of the total variance of the estimate that can be attributed to the missing data.