Missing Data Handout

PHS 2000B

April 2023

1 Missing Data Introduction

Missing data are a part of an empiricist's life. In fact, more often than not, you will find that the data you have collected or the data you have been given will have values for covariates that are missing or values for covariates that are purposely suppressed. Excluding observations with missing data from your analysis will almost certainly lead to inefficient estimates. Perhaps more troublingly, however, excluding observations with missing data may lead to biased estimates as well.

1.1 Missingness mechanisms

The statistical literature on missing data has typically focused on mechanisms which describe why the data are missing, i.e., **missingness mechanisms**. Little and Rubin (1987) classified missingness mechanisms into three categories:

1. Missing Completely At Random (MCAR)
2. Missing At Random (MAR)
3. Missing Not At Random (MNAR or NMAR)

1.2 Notation

Let Z describe a vector of all variables of interest in our analysis including the outcome, exposure, and all covariates. For a given individual i in our analysis, we can partition Z into two components, Z^{obs} and Z^{mis} where Z^{obs} are all variables that are observed while Z^{mis} are all variables that are missing. Based on Z^{obs} and Z^{mis}, we now define a vector R where $R = 1$ for all corresponding Z^{obs} and $R = 0$ for all corresponding Z^{mis}.

For example, let $Z = \{$ age, highest level of education, earnings $\}$. For an arbitrary individual i, let $Z_i = \{30, ., .\}$ where . indicates a missing value. Thus, for this individual i, $R = \{1, 0, 0\}$.

We can now define MCAR, MAR, and MNAR based on probability distributions for R. Suppose that θ represents a set of parameters that describes the probability distribution of R (e.g., mean, variance, etc.).

1.3 Missing Completely At Random (MCAR)

Data are defined as being MCAR if:

$$Pr(R|Z) = Pr(R|Z^{obs}, Z^{mis}) = Pr(R|\theta) \tag{1}$$

So, data are MCAR if the probability of being observed (and, consequently, the probability of being missing) is independent of values of Z that are observed or would have been observed. In other words, data are MCAR if the mechanism for missingness is random chance.

Consider an ideal randomized controlled trial (RCT) with a binary treatment. Recall that an ideal RCT has the following characteristics: 1) no issues with randomized treatment assignment; 2) well-defined treatment; 3) double blind; 4) perfect adherence; and 5) no loss to follow-up. In this study, the unobserved potential outcomes are MCAR since they are missing due to a random process.

1.4 Missing At Random (MAR)

Data are defined as being MAR if:

$$Pr(R|Z) = Pr(R|Z^{obs}, Z^{mis}) = Pr(R|Z^{obs}, \theta) \tag{2}$$

So, data are MAR if the probability of being observed (and, consequently, the probability of being missing) is a function of the observed values of Z. Alternatively, data are MAR if, conditional on the observed values of Z, the mechanism for missingness is random chance.

Consider a stratified, ideal RCT with binary treatment. Recall that a stratified RCT is a RCT where randomized treatment assignment occurs within pre-defined and known strata. In this study, the unobserved potential outcomes are MAR since they are missing due to a random process conditional on the strata.

1.5 Missing Not At Random (MNAR)

Data are defined as being MNAR if:

$$Pr(R|Z) = Pr(R|Z^{obs}, Z^{mis}, \theta) \tag{3}$$

So, data are MNAR if the probability of being observed (and, consequently, the probability of being missing) is a function of values of Z that would have been observed. Alternatively, data are MNAR if, conditional on the observed values of Z, the mechanism for missingness depends on the unobserved values of Z.

Consider an observational study where there is selection-on-unobservables (i.e., there are unmeasured confounders of the relationship between a treatment and outcome of interest). In this study, the unobserved potential outcomes are MNAR since they are missing due to unobserved factors.

2 Structural Representations

2.1 Selection Bias

2.1.1 Without Collider Bias

No Selection into the Study

The above DAG represents a familiar scenario, in which we have one measured confounder for the effect of A on Y. We can approach this DAG as we have in the past - condition on V to block the back door path $A \leftarrow V \rightarrow Y$.

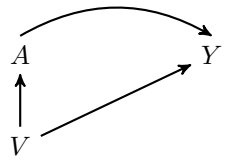


Figure 1. No Selection

The conditional effect under this scenario is represented by

$$E[Y|A = 1, V = v] - E[Y|A = 0, V = v]$$

There are three options for identifying the marginal effect:

1. **Standardize to the distribution of V in the population**

$$\sum_v (E[Y|A=1,V=v] - E[Y|A=0,V=v])Pr(V=v)$$

2. **Create inverse probability of treatment weights for all individuals in the population**

$$\text{For } A=1: \frac{1}{Pr[A=1|V=v]} \quad \text{For } A=0: \frac{1}{1-Pr[A=1|V=v]}$$

3. **Assume no EMM**

$$E[Y|A=1,V=v] - E[Y|A=0,V=v] = E[Y|A=1] - E[Y|A=0]$$

See Lab 1 and Identifying Causal Effects Handouts for more information.

Selection into the Study

Now consider a situation in which we have missing data. For the purpose of this example, we can think about if we had selected individuals only who had a certain characteristic, which we can think of as V since there is an arrow from V into R.

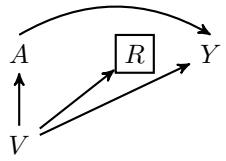


Figure 2. Selection Based on R

Now let's check if we can identify the conditional causal effect of A on Y. Again, we can condition on V to block the backdoor path. But now, we are also conditioning on R, specifically we are looking at the effect within the stratum $R=1$ because those are individuals for whom we have data available. This effect is identified by the expression

$$E[Y|A=1,V=v,R=1] - E[Y|A=0,V=v,R=1]$$

To determine if this effect is equivalent to the effect we identified above in the absence of selection, we can check whether $Y \perp\!\!\!\perp R$, either marginally or conditional on other covariates. This means we are checking whether there are any open paths from Y to our selection node R.

We can see that there is a path from $Y \leftarrow V \rightarrow R$ on the original DAG, but when we condition on V or A this path is blocked. Therefore, in this DAG we have a conditional independence $Y \perp\!\!\!\perp R|(V,A)$.

Recall $Pr(A|B) = Pr(A)$ and $Pr(A,B) = Pr(A)Pr(B)$ if $A \perp\!\!\!\perp$ B. So, using the rules of conditional probability and independence, we can re-write our conditional effect in the presence of selection bias, removing R from the conditioning statement. We can see now that the conditional effect under selection is equivalent to the conditional effect in the whole population.

$$E[Y|A=1,V=v,R=1] - E[Y|A=0,V=v,R=1] = E[Y|A=1,V=v] - E[Y|A=0,V=v]$$

If we were interested in the marginal effect for the whole population, but only have data on the subset for whom $R=1$, then we can check whether our usual tools work.
There are three options for identifying the marginal effect:

1. **Standardize to the distribution of V in the population**

$$\sum_v (E[Y|A=1,V=v,R=1] - E[Y|A=0,V=v,R=1])Pr(V=v|R=1)$$

For this to equal the marginal effect we previously identified, we need for both $Y \perp\!\!\!\perp R|(V,A)$ AND $V \perp\!\!\!\perp R$.

The reason we need both of these independences is because we are now standardizing to the distribution of V among those for whom we have data, i.e., using $Pr(V=v|R=1)$. Since V influences selection, $V \to R$, we know that $V \not\perp\!\!\!\perp R$. Therefore, $Pr(V=v|R=1) \neq Pr(V=v)$.

Even though we have an equal conditional effect, may not necessarily have an equal marginal effect, in the presence of effect modification since the effect would vary across strata V, and we would need to weight these effects by the proportion of individuals within each strata in the whole population, which we don't know.

2. **Create inverse probability of treatment weights for all individuals in the population**

$$\text{For } A=1: \frac{1}{Pr[A=1|V=v]} \quad \text{For } A=0: \frac{1}{1-Pr[A=1|V=v]}$$

A similar reason why standardization fails to identify the marginal effect in the presence of EMM applies here. The intuition behind inverse probability weights is that it will remove the arrow corresponding to the variables in the denominator, such that weights like $\frac{1}{Pr[A=1|V=v]}$ will remove an arrow from V to A. However, we would also need to create censoring weights for individuals who are missing to remove the arrow remaining from $V \to A$.

The problem here is that selection depends on V, so we can't create proper inverse probability weights, when we don't have the full distribution of V.

Identifying the marginal effect using IPW is also not possible in this scenario.

3. **Assume no EMM**

$$\begin{aligned} &E[Y|A=1,V=v,R=1] - E[Y|A=0,V=v,R=1] \\ &= E[Y|A=1,V=v] - E[Y|A=0,V=v] \\ &= E[Y|A=1] - E[Y|A=0] \end{aligned}$$

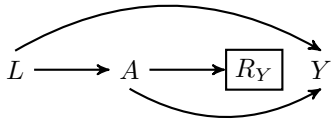
Since we know our conditional effect in our selected population is equal to the conditional effect in the target population, under the no EMM assumption this also is the marginal effect.

2.2 Missing Data

We can extend our structural representations of selection bias to representations of missing data on specific variables. Below we will review how to determine the type of missingness, and determine whether we can identify the casual effect of A on Y in the presence of missingness.

2.2.1 MAR Example

Let's consider a DAG where we assume that missingness in the outcome Y depends on the exposure A.



Step 1: Determine the missingness structure

We can check $R_Y \perp\!\!\!\perp \boldsymbol{Z^{mis}}|\boldsymbol{Z^{obs}}$ based on the variables for which we have observed data and missing data in DAG. Here, $\boldsymbol{Z^{obs}} = \{L, A\}$ and $\boldsymbol{Z^{mis}} = \{Y\}$.

Since we have conditional independence between our variable that has missingness (Y) and the missingness indicator (R_Y), data on Y are MAR.

Step 2: Address confounding bias by conditioning on variables to block back door paths between A and Y

To block the back door path, we can condition on L. This enables the identification of the conditional causal effect of A on Y, among those for whom data is available ($R = 1$):

$$E[Y|A = 1, L = l, R_Y = 1] - E[Y|A = 0, L = l, R_Y = 1]$$

Step 3: Asses for collider stratification bias (structural selection bias)

This is not a concern based on this DAG. Our missingness indicator variable R_Y is not a collider on *any* path since it only has one arrow going into it from A.

Step 4: Assess for failure to generalize bias.

1. Check $Y \perp\!\!\!\perp R_{miss}|Z^{obs}$ to determine whether conditional effect will be unbiased.

   We know that $Y \perp\!\!\!\perp R_Y|(L, A)$ because we used this independence to determine the missingness! Note that this is not always the case (see below example).

   $$E[Y|A = 1, L = l, R_Y = 1] - E[Y|A = 0, L = l, R_Y = 1] = E[Y|A = 1, L = l] - E[Y|A = 0, L = l]$$

2. Check whether the variable you need to condition on to identify the causal effect is **marginally** independent of the missingness variable, i.e., $L \perp\!\!\!\perp R_Y$

   We can see that $L \not\perp\!\!\!\perp R_Y$ because it has an indirect effect on R_Y through A. While conditioning on A would make L and R_Y independent, we need marginal independence to get $Pr(L = l|R = 1) = Pr(L = l)$.

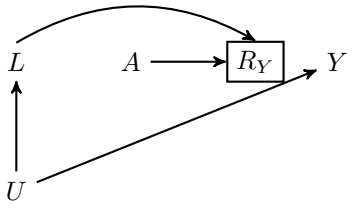
   $$\sum_v (E[Y|A = 1, L = l, R = 1] - E[Y|A = 0, L = l, R = 1])Pr(L = l|R = 1)$$
   $$\neq \sum_v (E[Y|A = 1, L = l] - E[Y|A = 0, L = l])Pr(L = l)$$

   Note: We previously proved that the inside of these equations are equal. The conditional effect in each stratum in the selected population, may be equal to the conditional effects in each stratum of the target population. However, if the distribution of L differs in the selected population from the distribution of L in the target population (i.e., $Pr(L = l|R = 1) \neq Pr(L = l)$) and there is EMM, then we cannot identify the marginal causal effect in the target population. The marginal causal effect estimate from our selected population will be a **biased estimate** of the target population marginal effect, *this is what we mean by failure to generalize!*

Only if we make the assumption that there is no EMM, then we can identify the marginal effect in the target population as equal to the conditional effect in the selected population.

2.2.2 Checking Missingness Representations

a)

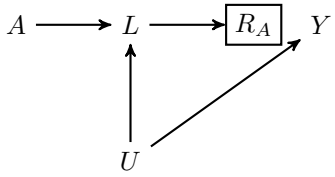


Data are MAR

$$R_Y \perp\!\!\!\perp Y|(A, L)$$

We find that if we condition on A and L, then there are no open paths from R_Y to Y

b)

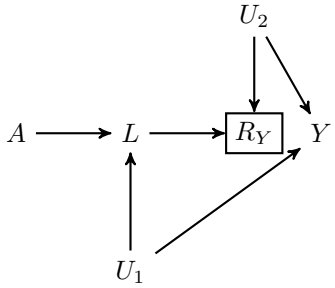


Data are MAR

$$R_A \perp\!\!\!\perp A|(L, Y)$$

Remember colliders are path-specific! We already have an open path from A to Y through U because we are conditioning on a descendent of a collider. Conditioning on L isn't a problem when checking whether $A \perp\!\!\!\perp R_A$ because on this path, L is not a collider.

c)

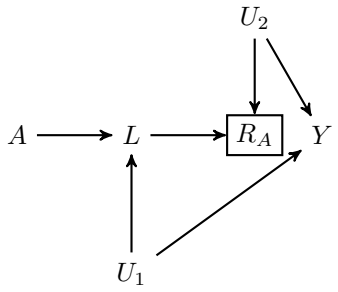


Data are MNAR

$$R_Y \not\perp\!\!\!\perp Y|(L, A)$$

Since we have another unmeasured variable, U_2, that influences both the missing data indicator R_Y, the data are now missing not at random. There is no way to block the path from Y to R_Y since there is an unmeasured common cause.

d)



Data are MNAR

$$R_A \not\perp\!\!\!\perp A|(L, Y)$$

While we had MAR in scenario c, this is now MNAR. Since we have to check whether $R_A \perp\!\!\!\perp A$ conditional on BOTH (L, Y), we now have a backdoor path open from $A \to [L] \leftarrow U_1 \to [Y] \to U_2 \leftarrow R_A$, because L and Y are both colliders on this path. Since there is no way to block the path from A to R_A by conditioning on observed variables alone.

Additional Concepts

Missingness pattern

Suppose you have a dataset with multiple variables that have missing values. The pattern of missing data is described as being **monotone** if we can rearrange the variables (i.e., columns) such that observing one variable X_b for an individual i implies that X_a is observed for the same individual (where $a < b$). Figure 1, adapted from Horton and Kleinman (2007), illustrates data that have monotone missingness (left panel) and non-monotone missingness (right panel). Green cells denote data that are observed while red denote data that are missing. Note that for the panel on the right, there is no way to re-arrange the columns such that the pattern of missing and non-missing observations look like the pattern in the panel on the left.

Figure 1: Missingness Pattern

Unit non-response vs. item non-response

Unit non-response is a situation where, for a given subject i, none of the variables are measured. As an example, you can think of a longitudinal dataset where, say, an individual was not followed up at an arbitrary time t but followed at all other time periods.

Item non-response is a situation where, for a given subject i, one or more variables are not measured but some variables are measured. As an example, you can think of a cross-sectional dataset where we may have information on an individual's age and gender but not highest level of education.

Non-ignorable vs. ignorable missingness

The notion of ignorability brings together ideas connected with both the parameters we are ultimately interested in estimating and the missingness mechanism. Specifically, missingness is said to be ignorable if the data are:

1. MCAR or MAR; and
2. Parameters of interest for our analysis (e.g., β) are distinct from parameters which govern the distribution of R

If missingness does not meet the two criteria above, then it is said to be non-ignorable. Thus, data that are MNAR are non-ignorable.

Multiple imputation as a method of addressing missing data

There are several methods of addressing missing data in our analyses. In lecture, we were introduced to four broad classes of methods:

1. Complete case analysis: Analyze only those data for which all variables are measured. This method is simple but leads to inefficiency and often to bias.
2. Imputation-based procedures: There are several imputation-based methods available. All of them involve trying to "fill in" the missing data using some pre-defined procedure. Today's lab will feature a discussion of multiple imputation (MI).
3. Weighting procedures: Weighted procedures involve weighted complete case analyses where the weights are defined as the inverse of the probability of being observed.
4. Model-based procedures: These procedures involve defining a model for the missing data and then conducting analyses and inferences based on that model. This method is transparent because readers know what the underlying model for missingness is that drives the analysis. A famous example from the Economics literature of a model-based procedure is the Heckman Selection Model.

Today's lab will focus on MI. Analysis using MI involves the following steps and is **conducted under the assumption that data are MAR.**

1. Construct m datasets where missing values have been replaced by simulated values based on some procedure. Note that non-missing values are the same as the original data across all m imputed datasets.
2. Conduct the analysis of interest across all m datasets. And,
3. Pool estimates across all m datasets and correctly estimate the variance around these estimates. Correctly estimating the variance requires incorporating information about both within-imputation variance and between-imputation variance.

Of these three steps, steps 2 and 3 are relatively straightforward. Step 2 is simple because it requires us to conduct the analysis we were planning to conduct across all m imputed datasets. Step 3 is simple because Rubin (1987) has helpfully provided us with a set of formulas to correctly calculate the point estimate of interest and its associated variance.

Step 1, however, is more nuanced because there are several considerations we have to make. We discuss these considerations in the following section.

Creating the multiply imputed datasets

How large should m be?

Rubin (1987) showed that the efficiency of a result from m imputations relative to an infinite number of imputations is a function of both m and $\hat{\lambda}$, where $\hat{\lambda}$ refers to the fraction of missing information about our estimates of interest. Importantly, Rubin also showed that a small number of imputations (m between 5 and 10) is usually sufficient to obtain relatively efficient estimates when using multiple imputation.

As a refresher, within the missing data world, there are two useful quantities that help us think about how much missingness is contributing to the relative inefficiency of our estimates. They are r, the relative increase in variance due to non-response, and $\hat{\lambda}$, the fraction of missing information.

$$r = \frac{(1 + m^{-1})B}{\bar{V_\beta}} \tag{1}$$

$$\hat{\lambda} = \frac{r + 2/(\nu + 3)}{r + 1} \tag{2}$$

Where B is the between-imputation variance, V_β is the within-imputation variance, and ν is the degrees of freedom we need to use for our inferences given m, B, and V_β. Note that at the limit of m, i.e., when $m \to \infty$, $r = \frac{B}{\bar{V_\beta}}$ and, consequently, $\hat{\lambda} = \frac{B}{T}$, where $T = B + \bar{V_\beta}$.

What should we include in the imputation model?

Meng (1994) showed that multiple imputation models should be at least as rich as the analytic model. This implies that we should include all variables we will use in our analysis plus other available covariates in our multiple imputation model.

What procedure should we follow to impute the missing data?

There are several methods available for imputing missing data. Often, the choice of the imputation method depends on the pattern of missingness as well as the nature of the variables to be imputed. In lab today, we will go over two popular methods of imputation:

1. Imputing missing data by assuming that the missing and non-missing data follow a specific joint distribution. This method is known as Joint Modeling (JM).
2. Imputing missing data on a variable-by-variable basis by specifying a set of conditional densities. This method is known as Fully Conditional Specification (FCS).

In terms of the first procedure, we will apply a multiple imputation program called Amelia II. In terms of the second procedure, we will use a program called Multiple Imputation with Chained Equations (MICE).

Amelia II

Amelia II is a multiple imputation package in R developed by Honaker, King, and Blackwell (Version 1.7.6; 2019).

The program imputes missing data under two key assumptions:

1. The complete data, i.e., the data consisting of both non-missing and missing values, follow a multivariate normal distribution. Specifically, for a complete dataset Z, $Z \sim \mathcal{N}(\mu, \Sigma)$ where μ refers to a vector of means of all variables in the data and Σ refers to the associated variance-covariance matrix.
2. The missingness mechanism is MAR.

The procedure that Amelia II uses to estimate parameters of the multivariate normal distribution is called the EMB algorithm (Expectation Maximization with bootstrap).

For a given dataset Z with n observations, Amelia II imputes data in three steps:

1. Draws a bootstrap sample of size n with replacement
2. Runs the Expectation Maximization algorithm in the bootstrap sample to estimate μ and Σ
3. Uses estimates of μ and Σ and the non-missing data to impute missing observations

Amelia II goes through these three steps m times to create m complete datasets. We can then combine the estimates and variances using Rubin's rules.

Multiple Imputation with Chained Equations (MICE)

MICE is a multiple imputation program developed by van Buuren and Groothuis-Oudshoorn (Version 2.9; 2011).

The program imputes missing data under the following assumptions:

1. The missingness mechanism is MAR.

The technique MICE uses to impute data is known as chained equations. Chained equations is a method whereby several equations are iteratively estimated and each equation uses results from the equation that was estimated prior to it. To put this more technically, chained equations is a procedure of iteratively estimating several univariate equations to estimate parameters of the joint distribution and use this information to impute the missing data. For details on the chained equations that are estimated in MICE, please see the lecture slides or van Buuren and Groothuis-Oudshoorn (Version 2.9; 2011).

How should we decide between using Ameila II or MICE?

There does not seem to be a great deal of guidance on when to use which procedure. Some argue that it is better to use Amelia II when we have confidence that the joint distribution is indeed multivariate normal. However, as the authors of Amelia II argue, even in instances where the multivariate normal distribution is a crude approximation of the true joint distribution, Amelia II "works well".

If you are concerned about the multivariate normal assumption, MICE can be an attractive option for imputation. MICE can also be an attractive procedure if you believe it is difficult to specify the joint distribution of the data more generally. However, as noted in lecture and by the authors of the program, MICE can lead to specifying chained equations for which no joint distribution exists. There do not seem to be theoretical results which tell us whether or not this issue poses any problems as far as the quality of the imputation is concerned. However, the authors of MICE note that some simulation work suggests that the incompatibility problem does not pose serious problems in practice.

Finally, when deciding between using the EM vs FCS algorithms (i.e., between using the Amelia II vs. MICE packages), note that you are more likely to experience model convergence issues and slow processing times using FCS compared to EM but will have more flexibility in modeling the missing variables.

NHANES data

We will be using the Amelia II and MICE multiple imputation procedures on NHANES data. NHANES, which stands for the National Health and Nutrition Examination Survey, is a survey conducted by the National Center for Health Statistics to study the health and nutritional status of adults and children in the United States. The NHANES surveys a nationally representative sample of 5000 individuals each year and collects information on demographic, socioeconomic, dietary, and health-related matters. The survey also records medical, dental, and physiological measurements and conducts lab tests as well.

R code

Let's start by installing and loading the relevant packages:

```
# Installing pacakages
if (!require("RNHANES")) install.packages("RNHANES")
if (!require("VIM")) install.packages("VIM")
if (!require("Amelia")) install.packages("Amelia")
if (!require("mitools")) install.packages("mitools")
if (!require("mix")) install.packages("mix")
if (!require("mice")) install.packages("mice")
if (!require("lattice")) install.packages("lattice")

# Loading packages
library("RNHANES")
library("VIM")
library("Amelia")
library("mitools")
library("mix")
library("mice")
library("lattice")
```

As mentioned above, we will be using data from NHANES (2011-12 survey) to investigate whether having eaten in the last 30 minutes is associated with systolic blood pressure values. The variables we plan on using for our analysis are:

- `RIAGENDR`: Gender
- `RIDAGEYR`: Age in years at screening

- RIDRETH1: Race/Hispanic origin
- DMDEDUC2: Education level (categorical)
- INDHHIN2: Household income (categorical)
- BPXSY2 : Systolic blood pressure
- BPQ150A : Having eaten in the last 30 minutes prior to blood pressure measurement (yes or no)

We will restrict our analysis to individuals aged 20 years or older.

Let's load the dataset from the RNHANES package and do some basic data cleaning.

```
# Loading the data
nhanes <- nhanes_load_data("BPX_G", "2011-2012", demographics = TRUE)

# Data cleaning steps

# Subsetting data to keep relevant variables
nhanes2 <- subset(nhanes, select = c(RIAGENDR,RIDAGEYR,RIDRETH1,DMDEDUC2,
                                     INDHHIN2,BPXSY2,BPQ150A))

# Drop subjects with age<20
nhanes3 <- nhanes2[!(nhanes2$RIDAGEYR<20),]

# Recode "Refused" (77) and "Don't know" (99) to missing for annual income
nhanes3$INDHHIN2[nhanes3$INDHHIN2==77] <- NA
nhanes3$INDHHIN2[nhanes3$INDHHIN2==99] <- NA

# Recode "Refused" (7) and "Don't know" (9) to missing for education
nhanes3$DMDEDUC2[nhanes3$DMDEDUC2==7] <- NA
nhanes3$DMDEDUC2[nhanes3$DMDEDUC2==9] <- NA

# Dichotomize household income into <20,000 (0) and >=20,000 (1)
nhanes3$HHINBIN <- ifelse((nhanes3$INDHHIN2<=4 | nhanes3$INDHHIN2==13),
                          ifelse(is.na(nhanes3$INDHHIN2), NA, 0), 1)

# Change Gender and Eating variables to binary 0/1 instead of 1/2
nhanes3$RIAGENDR[nhanes3$RIAGENDR == 2] <- 0
nhanes3$BPQ150A[nhanes3$BPQ150A == 2] <- 0

# Dropping original household income data from our analytic dataset
nhanes4 <- subset(nhanes3, select = -c(INDHHIN2))
```

Part 1. Visualizing Missingness

Burton and Altman (2004) recommend quantifying the completeness of the analytic dataset as the first step in reporting analyses with missing data. Let us therefore start our analysis by investigating the magnitude of missingness in our dataset. We can first just calculate the percent missing for each variable.

```
# Creating a function to calculate percent missing
pMiss <- function(x){sum(is.na(x))/length(x)*100}

# Applying function to analytic dataset to quantify percent missing across all variables
round(apply(nhanes4,2,pMiss), 2)
```

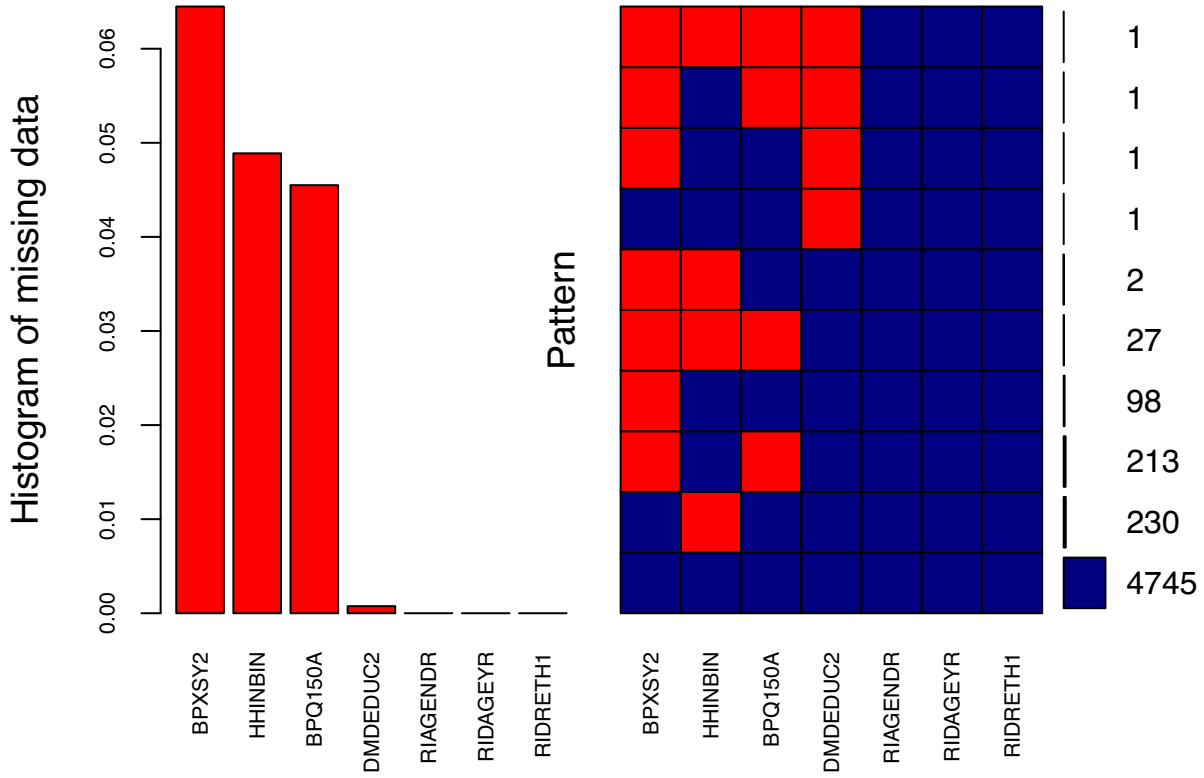
```
## RIAGENDR RIDAGEYR RIDRETH1 DMDEDUC2   BPXSY2  BPQ150A  HHINBIN
##     0.00     0.00     0.00     0.08     6.45     4.55     4.89
```

We can see that some of our variables (gender, age, and race/ethnicity) are complete and that there is fairly

low levels of missingness for all other variables (<10%).

Now, let's visualize the pattern of missingness using the "VIM" package. Specifically, we are going to use the `aggr` command, which allows us to plot the amount of missing data in the variables in our dataset.

```
# Visualize missingness
# Check out ?aggr to look at what the various options mean
aggr_plot <- aggr(nhanes4, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,
                  labels=names(data), cex.axis=0.7, gap=1, prop=c(TRUE,FALSE),
                  ylab=c("Histogram of missing data","Pattern"))
```




```
##
##  Variables sorted by number of missings:
##  Variable        Count
##    BPXSY2 0.0644858056
##   HHINBIN 0.0488813687
##   BPQ150A 0.0454972739
##  DMDEDUC2 0.0007520211
##  RIAGENDR 0.0000000000
##  RIDAGEYR 0.0000000000
##  RIDRETH1 0.0000000000
```

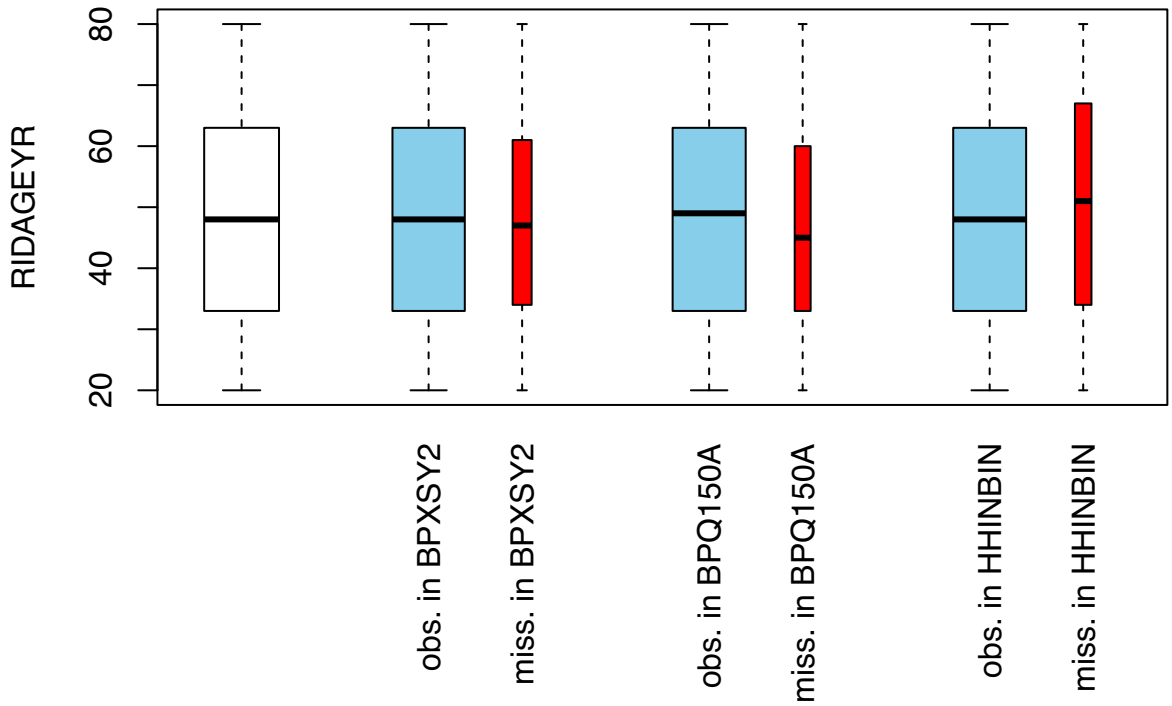
We can see from this figure that 4,745 observations have no missing values (out of 5319=89%) and that blood pressure and responding to the question about having eaten are generally missing for the same observations, while income is generally missing for different observations.

We should also check for predictors of missingness. One such helpful plot to look at this is shown below to see if age is related to missingness.

```r
# Visualize predictors of missingness
# Selecting variables to visualize
vars <- c("RIDAGEYR","BPXSY2", "BPQ150A", "HHINBIN")

# Visualizing by checking missingness of three variables against age
pbox(nhanes4[,vars])
```

```
## Warning in createPlot(main, sub, xlab, ylab, labels, ca$at): not enough space
## to display frequencies
```



From this plot, we can see that individuals missing blood pressure values and information about whether they ate recently are somewhat younger and those who didn't share their income are somewhat older.

Part 2. Multiple Imputation using AMELIA package

Recall that Amelia II imputes data under two assumptions:

1. The complete data, i.e., the data consisting of both non-missing and missing values, follow a multivariate normal distribution.
2. The missingness mechanism is MAR.

Let's apply Amelia II to a situation where we have only one variable with missing data and another situation where we have multiple variables with missing data.

Example 1: Only one missing variable

First, let's look at an example where we want to investigate the relationship between blood pressure (the outcome) and age (the "exposure"). Recall that age has no missing data but a few observations are missing for blood pressure. Let's begin by imputing values of the blood pressure variable. We will then estimate a simple linear regression of blood pressure on age across all imputed datasets. Finally, we will pool the estimates and variances using Rubin's rules.

Imputing the missing value Remember that we want our imputation model to be at least as rich as our outcome model. In this analysis, we are interested in estimating a linear regression of blood pressure on age. However, to impute missing values in the blood pressure variable, we will use information on age, gender, and race/ethnicity.

```
# Subsetting data for imputation purposes: selecting age, gender, race/ethnicity,
# and blood pressure
reduced <- subset(nhanes4, select = c(RIAGENDR, RIDAGEYR, RIDRETH1, BPXSY2))

# Creating 5 imputed datasets where we impute missing values of blood pressure
a.out <- amelia(reduced, m = 5)
```

```
## -- Imputation 1 --
##
##   1  2
##
## -- Imputation 2 --
##
##   1  2
##
## -- Imputation 3 --
##
##   1  2
##
## -- Imputation 4 --
##
##   1  2
##
## -- Imputation 5 --
##
##   1  2
```

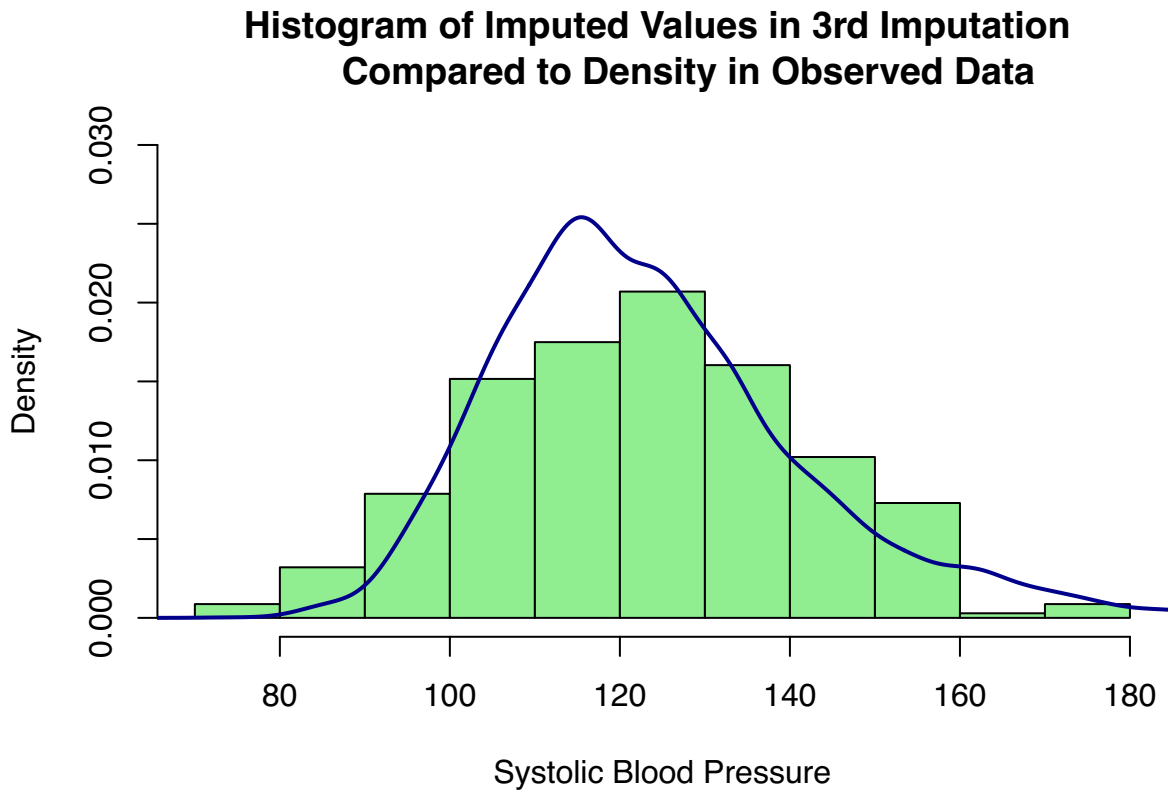
The Amelia II object `a.out` contains information from our five imputations. It is always a good idea to check to see if our imputed data has reasonable values. One way we can do this is by plotting the distribution of the imputed data against the distribution of the observed values of the data. Ideally, we'd want the distributions to look similar. Let's do this for our third imputed dataset. Results should be similar across the remaining four imputed datasets as well.

```
# Choosing third imputed dataset and extracting the imputed blood pressure variable
i=3
one_imp <- a.out$imputations[[i]]$BPXSY2 # stores the imputed values

# Extracting the observed values of the blood pressure variable
obs_data <- nhanes4$BPXSY2 # stores the observed data

# Creating a histogram of the imputed variable
# and superimposing the density of the observed variable
hist(one_imp[is.na(obs_data)], prob=TRUE, xlab="Systolic Blood Pressure",
```

```
     main="Histogram of Imputed Values in 3rd Imputation
     Compared to Density in Observed Data",
     col="lightgreen", ylim=c(0,0.03))
lines(density(obs_data[!is.na(obs_data)]), col="darkblue", lwd=2)
```



Pooling estimates The Amelia II package itself doesn't have a way to combine the estimates from these imputed datasets based on Rubin's rules, but there are a number of other packages that can do so. In the code below, we fit our outcome regression model of interest (an OLS model predicting blood pressure based on age) in each of the imputed datasets using the *lapply* function and then use the *mitools* package to calculate pooled estimates and variances.

```
# Use lapply to run linear regression models in each imputed dataset
outimp <- lapply(a.out$imputations, function( x ){lm(BPXSY2 ~ RIDAGEYR, data=x)})

# Use mitools to extract coefficients and variance estimates and then use Rubin's
# rules to pool/combine these estimates
betas <- MIextract(outimp, fun = coef)
vars <- MIextract(outimp, fun = vcov )
summary(MIcombine(betas, vars))
```

```
## Multiple imputation results:
##       MIcombine.default(betas, vars)
##                 results         se     (lower      upper) missInfo
## (Intercept) 100.9339113 0.69426824 99.5714397 102.2963828      7 %
## RIDAGEYR      0.4641386 0.01326799  0.4381149   0.4901624      5 %
```

Some points to keep in mind:

- The imputation model included gender and race/ethnicity but these weren't included in the outcome regression model. The imputation model should include any important predictors of missing values as well as all variables included in the outcome regression model.
- While this example uses OLS for the outcome model, we could use any other outcome regression model here as well, changing "lm" to "glm" and setting a link function

Example 2: Multiple missing variables

Now let's impute all of the missing variables in our dataset to look at the question of whether having eaten recently predicts blood pressure. Note that we have some categorical variables (race/ethnicity and education) and some dichotomous variables (household income and having eaten in the last 30 minutes). The recommended approach is to create dummy variables for categorical variables and to predict these dichotomous variables using the multivariate normality assumption, even though you will end up with predicted values that are not whole numbers. It is possible to round these numbers after imputation, but it will lead to biased estimates and is not recommended.

Imputing the missing value Let's begin by creating dummy variables for all the categorical variables in our data and by subsetting the data to create the imputation dataset.

```
# Creating dummy variables for race
nhanes4$race.ind2 <- ifelse((nhanes4$RIDRETH1==2),
                            ifelse(is.na(nhanes4$RIDRETH1), NA, 1), 0)
nhanes4$race.ind3 <- ifelse((nhanes4$RIDRETH1==3),
                            ifelse(is.na(nhanes4$RIDRETH1), NA, 1), 0)
nhanes4$race.ind4 <- ifelse((nhanes4$RIDRETH1==4),
                            ifelse(is.na(nhanes4$RIDRETH1), NA, 1), 0)
nhanes4$race.ind5 <- ifelse((nhanes4$RIDRETH1==5),
                            ifelse(is.na(nhanes4$RIDRETH1), NA, 1), 0)

# Creating dummy variables for education
nhanes4$educ.ind2 <- ifelse((nhanes4$DMDEDUC2==2),
                            ifelse(is.na(nhanes4$DMDEDUC2), NA, 1), 0)
nhanes4$educ.ind3 <- ifelse((nhanes4$DMDEDUC2==3),
                            ifelse(is.na(nhanes4$DMDEDUC2), NA, 1), 0)
nhanes4$educ.ind4 <- ifelse((nhanes4$DMDEDUC2==4),
                            ifelse(is.na(nhanes4$DMDEDUC2), NA, 1), 0)
nhanes4$educ.ind5 <- ifelse((nhanes4$DMDEDUC2==5),
                            ifelse(is.na(nhanes4$DMDEDUC2), NA, 1), 0)

# Subsetting data
reduced2 <- subset(nhanes4, select = -c(RIDRETH1, DMDEDUC2))
```

Now, let's go ahead and use Amelia II to impute the missing values in our data. As in the previous example, we will also check to see if the imputed values make sense. In this example, we will investigate the distribution of the household income variable in the imputed dataset against the same variable in the non-imputed dataset.

```
# Run the Amelia imputation algorithm and creating 5 imputed datasets
a.out2 <- amelia(reduced2, m = 5)
```

```
## -- Imputation 1 --
##
##   1  2  3
##
## -- Imputation 2 --
```

```
##
##   1  2  3
##
## -- Imputation 3 --
##
##   1  2  3
##
## -- Imputation 4 --
##
##   1  2  3  4
##
## -- Imputation 5 --
##
##   1  2  3
```

```
# Comparing the mean of the household income variable in the imputed data vs.
# non-imputed data. We will check this in the third imputed dataset.

# Extracting household income variable from third imputed dataset
i <- 3
one_imp <- a.out2$imputations[[i]]$HHINBIN # stores the imputed values

# Extracting household income variable from the non-imputed dataset
obs_data <- reduced2$HHINBIN # stores the observed values

# Checking the distribution
summary(one_imp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.2249  0.4852  1.0000  0.7505  1.0000  2.0175
```

```
summary(obs_data[!is.na(obs_data)])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  1.0000  1.0000  0.7507  1.0000  1.0000
```

Pooling estimates Now, let's fit a linear regression model of blood pressure on having eaten in the 30 minutes prior to blood pressure measurement. In this model, we will control for gender, age, race, education, and household income. As before, we are going to estimate this model in all five imputed datasets and then combine the point estimates and variances using Rubin's rule.

```
# Use lapply to run linear regression models in each imputed dataset
outimp2 <- lapply(a.out2$imputations,
                  function( x ){lm(BPXSY2 ~ BPQ150A + RIAGENDR + RIDAGEYR + race.ind2 +
                                   race.ind3 + race.ind4 + race.ind5 + educ.ind2 +
                                   educ.ind3 + educ.ind4 + educ.ind5 + HHINBIN, data=x)})

# Use mitools to extract coefficients and variance estimates and then use Rubin's
# rules to pool/combine these estimates
betas <- MIextract(outimp2, fun = coef)
vars <- MIextract(outimp2, fun = vcov )
summary(MIcombine(betas, vars))
```

```
## Multiple imputation results:
##       MIcombine.default(betas, vars)
##                 results         se     (lower      upper) missInfo
```

```
## (Intercept) 100.1840078 1.29639100 97.6365282 102.7314873     10 %
## BPQ150A      -0.8378167 0.57755303 -1.9759834   0.3003500     14 %
## RIAGENDR      3.7221597 0.47505953  2.7884900   4.6558294     10 %
## RIDAGEYR      0.4537054 0.01389114  0.4264175   0.4809933      9 %
## race.ind2     0.3313479 1.03153072 -1.6908947   2.3535905      3 %
## race.ind3     0.8961433 0.86470407 -0.7986690   2.5909555      1 %
## race.ind4     5.6768622 0.89594085  3.9202801   7.4334444      3 %
## race.ind5     1.0508935 0.97214507 -0.8562281   2.9580151      6 %
## educ.ind2    -0.1523090 1.09360685 -2.3255960   2.0209780     23 %
## educ.ind3    -0.4065381 0.98876456 -2.3538531   1.5407768     13 %
## educ.ind4    -0.5752771 1.07899258 -2.7450236   1.5944693     32 %
## educ.ind5    -3.6409391 1.05467732 -5.7324807  -1.5493976     21 %
## HHINBIN      -1.6730704 0.59108974 -2.8393194  -0.5068215     16 %
```

We can also calculate the "r" and "$\hat{\lambda}$" values we discussed in class using the *mix* package. Recall that "r" refers to the relative increase in variance due to non-responses. That is, "r" is a statistic that tells us how much more our variance in our estimate of interest is inflated because of missingness in the data. Similarly, recall that "$\hat{\lambda}$" refers to the fraction of missing information about the estimate of interest.

```
# Feed the mi.inference function your beta and se estimates from the individual
# imputation models and it will report back the df, r, and lambda estimates
se <- MIextract(outimp2, fun=function(x){sqrt(diag(vcov(x)))})
as.data.frame(mi.inference(betas, se))
```

```
##                      est    std.err          df       signif      lower
## (Intercept) 100.1840078 1.29639100   467.16437 0.000000e+00 97.6365282
## BPQ150A      -0.8378167 0.57755303   222.76301 1.482891e-01 -1.9759834
## RIAGENDR      3.7221597 0.47505953   439.67657 3.552714e-14  2.7884900
## RIDAGEYR      0.4537054 0.01389114   534.82992 0.000000e+00  0.4264175
## race.ind2     0.3313479 1.03153072  5103.99421 7.480571e-01 -1.6908947
## race.ind3     0.8961433 0.86470407 87595.60236 3.000379e-01 -0.7986690
## race.ind4     5.6768622 0.89594085  3727.68874 2.635732e-10  3.9202801
## race.ind5     1.0508935 0.97214507  1317.32438 2.798928e-01 -0.8562281
## educ.ind2    -0.1523090 1.09360685    88.07795 8.895531e-01 -2.3255960
## educ.ind3    -0.4065381 0.98876456   251.46773 6.813071e-01 -2.3538531
## educ.ind4    -0.5752771 1.07899258    47.75594 5.963945e-01 -2.7450236
## educ.ind5    -3.6409391 1.05467732   103.67640 8.060834e-04 -5.7324807
## HHINBIN      -1.6730704 0.59108974   182.48652 5.168882e-03 -2.8393194
##                    upper           r       fminf
## (Intercept) 102.7314873 0.101968058 0.096392893
## BPQ150A       0.3003500 0.154735916 0.141672880
## RIAGENDR      4.6558294 0.105438152 0.099468363
## RIDAGEYR      0.4809933 0.094668327 0.089878342
## race.ind2     2.3535905 0.028800913 0.028375298
## race.ind3     2.5909555 0.006803518 0.006780221
## race.ind4     7.4334444 0.033866848 0.033275990
## race.ind5     2.9580151 0.058317601 0.056535373
## educ.ind2     2.0209780 0.270819772 0.230385922
## educ.ind3     1.5407768 0.144323739 0.132989709
## educ.ind4     1.5944693 0.407284932 0.317412045
## educ.ind5    -1.5493976 0.244434210 0.211487677
## HHINBIN      -0.5068215 0.173780710 0.157238199
```

- df: degrees of freedom associated with the t reference distribution
- r: relative increase in variance due to nonresponse
- fminf: fraction of missing information ($\hat{\lambda}$)

Part 3. Multiple Imputation using MICE package

Now let's try the MICE (Mulivariate Imputation with Chained Equations) package which uses Fully Conditional Specification. With MICE, you'll have the choice of a number of different methods to model the missing values.

```
methods(mice)
```

```
##  [1] mice.impute.2l.bin              mice.impute.2l.lmer
##  [3] mice.impute.2l.norm             mice.impute.2l.pan
##  [5] mice.impute.2lonly.mean         mice.impute.2lonly.norm
##  [7] mice.impute.2lonly.pmm          mice.impute.cart
##  [9] mice.impute.jomoImpute          mice.impute.lasso.logreg
## [11] mice.impute.lasso.norm          mice.impute.lasso.select.logreg
## [13] mice.impute.lasso.select.norm   mice.impute.lda
## [15] mice.impute.logreg              mice.impute.logreg.boot
## [17] mice.impute.mean                mice.impute.midastouch
## [19] mice.impute.mnar.logreg         mice.impute.mnar.norm
## [21] mice.impute.mpmm                mice.impute.norm
## [23] mice.impute.norm.boot           mice.impute.norm.nob
## [25] mice.impute.norm.predict        mice.impute.panImpute
## [27] mice.impute.passive             mice.impute.pmm
## [29] mice.impute.polr                mice.impute.polyreg
## [31] mice.impute.quadratic           mice.impute.rf
## [33] mice.impute.ri                  mice.impute.sample
## [35] mice.mids                       mice.theme
## see '?methods' for accessing help and source code
```

Some of the most commonly used are:

- *norm*: linear regression imputation (numeric data)
- *pmm*: predictive mean matching (numeric data that isn't normally distributed)
- *logreg*: logistic regression imputation (binary data)
- *polyreg*: multinomial/polytomous regression imputation (unordered categorical data)
- *polr*: ordinal/proportional odds regression imputation (ordered data with > 2 levels)

Let's repeat the example of imputing multiple variables with MICE.

Example 3: Repeat Example 2 with MICE

In this example, we will impute the continuous variable (blood pressure) using predictive mean matching, the dichotomous variables (having eaten in the last 30 minutes and income) using logistic regression, and the categorical variable (education) using a regression for oridinal data.

```
# Defining factor variables
nhanes4$race.f <- factor(nhanes4$RIDRETH1)
nhanes4$educ.f <- factor(nhanes4$DMDEDUC2)
nhanes4$income.f <- factor(nhanes4$HHINBIN)
nhanes4$eaten.f <- factor(nhanes4$BPQ150A)

# Subset to variables of interest and order from least to most missing
nhanesfactor <- subset(nhanes4, select = c(RIAGENDR, RIDAGEYR, race.f,
                                           educ.f, eaten.f, income.f, BPXSY2))

# Let's quickly see how our subsetted dataset ir ordered
```

```r
# Investigating the structure of our subsetted data
str(nhanesfactor)
```

Imputation model

```
## 'data.frame':    5319 obs. of  7 variables:
##  $ RIAGENDR: num  1 0 1 0 1 0 1 1 1 1 ...
##  $ RIDAGEYR: num  22 44 21 43 80 34 51 80 55 35 ...
##  $ race.f  : Factor w/ 5 levels "1","2","3","4",..: 3 3 5 4 3 3 5 3 5 3 ...
##  $ educ.f  : Factor w/ 5 levels "1","2","3","4",..: 3 4 3 3 5 5 3 3 5 5 ...
##  $ eaten.f : Factor w/ 2 levels "0","1": 1 2 1 1 1 2 2 1 1 1 ...
##  $ income.f: Factor w/ 2 levels "0","1": 2 2 1 2 2 2 NA 1 2 2 ...
##  $ BPXSY2  : num  104 118 126 102 96 114 144 124 124 108 ...
```

```r
# Investingating the proportion missing in our subsetted data
round(apply(nhanesfactor, 2, pMiss), 2)
```

```
## RIAGENDR RIDAGEYR   race.f   educ.f  eaten.f income.f   BPXSY2
##     0.00     0.00     0.00     0.08     4.55     4.89     6.45
```

```r
# Creating five imputed datasets using MICE
# Note that method = "" implies that the variable does not have any missing values
# Note also that the default method in MICE is predictive mean matching
impdata1 <- mice(nhanesfactor, m=5,
                 meth=c("", "",  "", "polr", "logreg", "logreg", "pmm"),
                 seed=500)
```

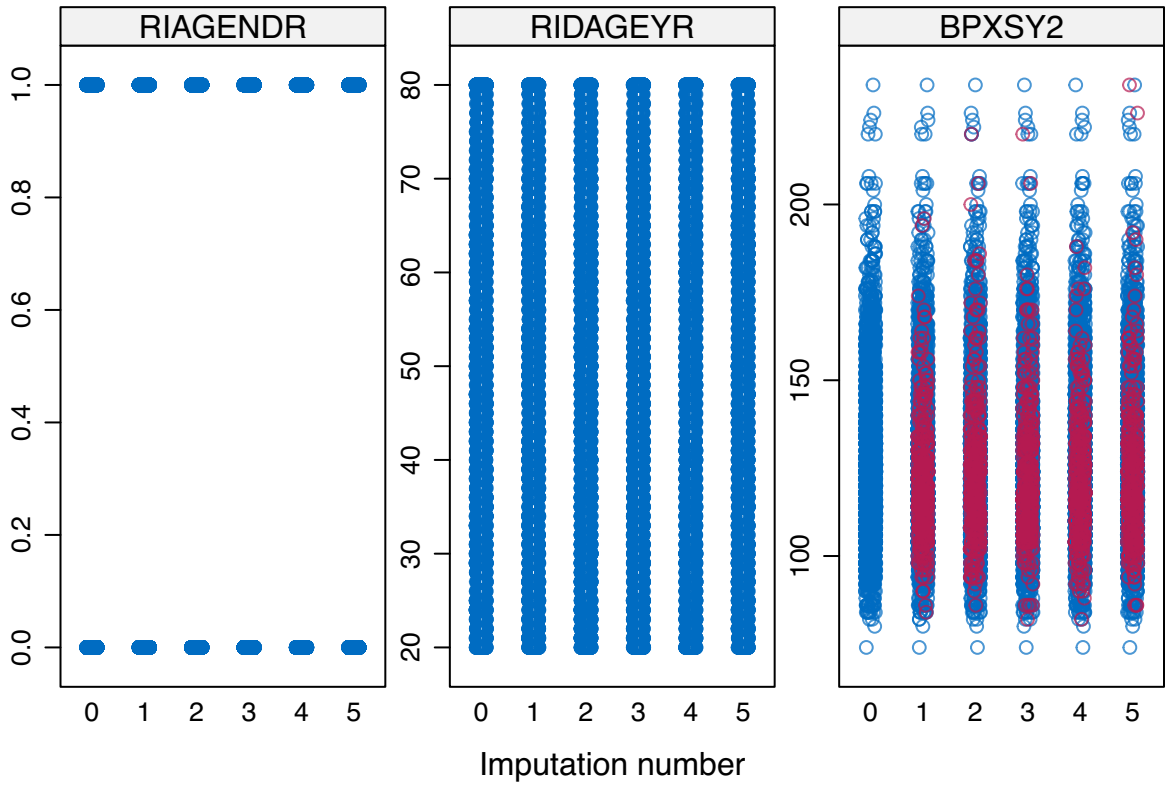
```
##
##  iter imp variable
##   1   1  educ.f  eaten.f  income.f  BPXSY2
##   1   2  educ.f  eaten.f  income.f  BPXSY2
##   1   3  educ.f  eaten.f  income.f  BPXSY2
##   1   4  educ.f  eaten.f  income.f  BPXSY2
##   1   5  educ.f  eaten.f  income.f  BPXSY2
##   2   1  educ.f  eaten.f  income.f  BPXSY2
##   2   2  educ.f  eaten.f  income.f  BPXSY2
##   2   3  educ.f  eaten.f  income.f  BPXSY2
##   2   4  educ.f  eaten.f  income.f  BPXSY2
##   2   5  educ.f  eaten.f  income.f  BPXSY2
##   3   1  educ.f  eaten.f  income.f  BPXSY2
##   3   2  educ.f  eaten.f  income.f  BPXSY2
##   3   3  educ.f  eaten.f  income.f  BPXSY2
##   3   4  educ.f  eaten.f  income.f  BPXSY2
##   3   5  educ.f  eaten.f  income.f  BPXSY2
##   4   1  educ.f  eaten.f  income.f  BPXSY2
##   4   2  educ.f  eaten.f  income.f  BPXSY2
##   4   3  educ.f  eaten.f  income.f  BPXSY2
##   4   4  educ.f  eaten.f  income.f  BPXSY2
##   4   5  educ.f  eaten.f  income.f  BPXSY2
##   5   1  educ.f  eaten.f  income.f  BPXSY2
##   5   2  educ.f  eaten.f  income.f  BPXSY2
##   5   3  educ.f  eaten.f  income.f  BPXSY2
##   5   4  educ.f  eaten.f  income.f  BPXSY2
##   5   5  educ.f  eaten.f  income.f  BPXSY2
```

```r
# Let's look at some useful information about our imputation
summary(impdata1)
```

```
## Class: mids
## Number of multiple imputations:  5
## Imputation methods:
## RIAGENDR RIDAGEYR   race.f   educ.f  eaten.f income.f   BPXSY2
##       ""       ""       ""   "polr" "logreg" "logreg"    "pmm"
## PredictorMatrix:
##          RIAGENDR RIDAGEYR race.f educ.f eaten.f income.f BPXSY2
## RIAGENDR        0        1      1      1       1        1      1
## RIDAGEYR        1        0      1      1       1        1      1
## race.f          1        1      0      1       1        1      1
## educ.f          1        1      1      0       1        1      1
## eaten.f         1        1      1      1       0        1      1
## income.f        1        1      1      1       1        0      1
```

```
# Let's also try to visualize the imputed data vs. observed data
# For now, let's use the "stripplot" function which comes with the "lattice" package
# This method only works for continuous variables
# Blue dots = observed values; red dots = imputed values

stripplot(impdata1, xlab="Imputation number" )
```




```
# Let's also extract the third imputed dataset and investigate imputed blood pressure
# Against the original blood pressure variable

# We use the "complete" command to extract a fully imputed dataset (here, dataset #3)
```

```r
test <- complete(impdata1, 3)

# Summarizing the blood pressure variable
summary(test$BPXSY2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    74.0   110.0   120.0   123.5   134.0   234.0
```

```r
summary(nhanesfactor$BPXSY2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    74.0   110.0   120.0   123.6   134.0   234.0     343
```

We can see the imputation process appeared to give a similar distribution of blood pressure as observed.

Pooling estimates Unlike Amelia II, MICE allows us to directly obtain the pooled estimates as follows:

```r
# Run the outcome regression model
MICEresults1 <- with(impdata1, lm(BPXSY2 ~ eaten.f + RIAGENDR + RIDAGEYR + race.f + educ.f
                        + income.f))

# Pooling estimates and summarizing results
summary(pool(MICEresults1))
```

```
##           term    estimate  std.error  statistic        df       p.value
## 1  (Intercept) 100.2043625 1.29486248 77.3861040  418.1481 6.143557e-250
## 2     eaten.f1  -0.7727465 0.55189379 -1.4001725 1146.9876  1.617321e-01
## 3     RIAGENDR   3.7731314 0.46211616  8.1648982 1382.1693  7.176238e-16
## 4     RIDAGEYR   0.4510069 0.01336982 33.7332048 3916.2717 3.215170e-219
## 5      race.f2   0.5021247 1.04781224  0.4792125  913.7400  6.319021e-01
## 6      race.f3   1.0101792 0.87985627  1.1481185 1526.9756  2.510995e-01
## 7      race.f4   5.8572284 0.88782820  6.5972544 3791.5881  4.768012e-11
## 8      race.f5   1.1956171 0.98898202  1.2089372  460.2429  2.273077e-01
## 9      educ.f2  -0.2766634 0.98212513 -0.2816987 2677.0117  7.781963e-01
## 10     educ.f3  -0.5229969 0.96819319 -0.5401783  438.8844  5.893481e-01
## 11     educ.f4  -0.5747884 0.93131625 -0.6171786 1273.6779  5.372273e-01
## 12     educ.f5  -3.6699619 0.95450945 -3.8448670 3263.5382  1.229193e-04
## 13   income.f1  -1.7333690 0.58970569 -2.9393798  168.0392  3.751712e-03
```

We obtained fairly similar estimates of the effect of having eaten in the last 30 minutes on blood pressure.

We can also get the r values for each variable. Recall that r refers to the relative increase in variance due to non-response and is specific to each estimated coefficient.

```r
# Calculate "r"
n <- rownames(summary(pool(MICEresults1)))
cbind(n, pool(MICEresults1)$pooled$riv)
```

```
##       n
##  [1,] "1"  "0.103090880192114"
##  [2,] "2"  "0.0547235763593256"
##  [3,] "3"  "0.0480697650810495"
##  [4,] "4"  "0.0162276597971458"
##  [5,] "5"  "0.0635984071343146"
##  [6,] "6"  "0.0447118387317803"
##  [7,] "7"  "0.0172575734407219"
##  [8,] "8"  "0.0973052577215748"
```

```
##  [9,] "9"  "0.0275583544784074"
## [10,] "10" "0.100132313792356"
## [11,] "11" "0.0509232813850506"
## [12,] "12" "0.0217975399314033"
## [13,] "13" "0.178380944082398"
```

Example 4: Imputing an interaction term

One point of confusion in multiple imputation is what to do with interaction terms or non-linear terms (e.g., squared terms) with missing data. Imputing variables and then calculating the interaction or non-linear term ("impute then transform" or "passive imputation") would be the logical choice but leads to biased estimates. One recommended and straightforward approach is to treat the interaction or non-linear term as "Just Another Variable". We'll look at the interaction between gender and having eaten in the last 30 minutes using this approach.

```
# Create the interaction variable as a factor variable
nhanes4$genderxeaten.f <- factor(nhanes4$RIAGENDR * nhanes4$BPQ150A)

# Subsetting data
nhanesfactor2 <- subset(nhanes4, select = c(RIAGENDR, RIDAGEYR, race.f,
                                            educ.f, eaten.f, genderxeaten.f,
                                            income.f, BPXSY2))

# Run the MICE imputation algorithm, modeling the interaction term with
# multinomial regression
impdata2 <- mice(nhanesfactor2, m=5,
                 meth=c("", "",  "", "polr", "logreg", "polyreg", "logreg", "pmm"),
                 seed=500)
```

```
##
##  iter imp variable
##   1   1  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   1   2  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   1   3  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   1   4  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   1   5  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   2   1  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   2   2  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   2   3  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   2   4  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   2   5  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   3   1  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   3   2  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   3   3  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   3   4  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   3   5  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   4   1  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   4   2  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   4   3  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   4   4  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   4   5  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   5   1  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   5   2  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   5   3  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
##   5   4  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
```

```
##   5   5  educ.f  eaten.f  genderxeaten.f  income.f  BPXSY2
```

```
# Run the outcome regression model
MICEresults2 <- with(impdata2, lm(BPXSY2 ~ eaten.f + genderxeaten.f + RIAGENDR + RIDAGEYR
                                  + race.f + educ.f + income.f))

# Pool estimates
summary(pool(MICEresults2))
```

```
##              term    estimate  std.error  statistic         df       p.value
## 1      (Intercept) 100.2746365 1.31157071 76.4538548  326.02421 2.954789e-210
## 2         eaten.f1  -1.0724972 0.80794641 -1.3274360  167.05285  1.861757e-01
## 3  genderxeaten.f1   0.3287172 1.13158723  0.2904921  383.14207  7.715969e-01
## 4         RIAGENDR   3.5581791 0.53947542  6.5956279  394.77543  1.362769e-10
## 5         RIDAGEYR   0.4520150 0.01359599 33.2462010 1257.63549 1.923901e-174
## 6          race.f2   0.3508417 1.03751702  0.3381552 1580.10356  7.352912e-01
## 7          race.f3   0.9235694 0.86455196  1.0682637 4757.29105  2.854558e-01
## 8          race.f4   5.7839720 0.88738431  6.5180011 3805.42656  8.059182e-11
## 9          race.f5   1.2009911 0.96622811  1.2429685 1388.71546  2.140892e-01
## 10         educ.f2  -0.1715827 1.05323400 -0.1629103  161.02836  8.707933e-01
## 11         educ.f3  -0.5019074 1.03327411 -0.4857447   94.86765  6.282682e-01
## 12         educ.f4  -0.5959955 1.03762488 -0.5743843   71.86190  5.675015e-01
## 13         educ.f5  -3.7614937 0.98420608 -3.8218558  559.18660  1.473157e-04
## 14       income.f1  -1.5874002 0.59790267 -2.6549476  131.45907  8.912314e-03
```

Some Practical Considerations for Multiple Imputation:

1. Remember that the imputation model must be *at least as complicated* as the outcome regression model. It must include all the predictors included in the outcome regression model but can and should contain additional variables that predict missing values.

2. As discussed in class, you should use a sufficient number of imputations, but will have diminishing returns at some point. You will need more imputations if you have a smaller effect estimate or more missing data. The 5 imputations used in this exercise may not have been sufficient for a publishable analysis.

3. When deciding between using Amelia vs MICE for multiple imputation, note that you are more likely to experience model convergence issues and slow processing times using the fully conditional approach (i.e., MICE) compared to the expectation maximization approach (i.e., Amelia), but MICE provides more flexibility in modeling the missing variables. The assumption of multivariate normality which underlies the Amelia package seems questionable in most common settings; however, in practice the methods often yield similar results.

4. In MICE, the ordering of variables can make a difference in results and model convergence. It is useful to try to order the variables logically, from least to most missing values. You can additionally change the number of iterations performed and the starting iteration which can also sometimes help with convergence.