Statistical Analyses with Missing Data

Day 3

Bryan Shepherd and Cindy Chen

Vanderbilt-Nigeria Biostatistics Workshop

June 2–6, 2025

Overview

Why Do We Care About Missing Data?

- Common in real-world data collection
- Impacts:
  - Statistical power
  - Bias
  - Validity of inference

Examples of Missing Data Scenarios

- Surveys with skipped items
- EHRs missing lab results
- Dropout in longitudinal studies

Types of Missingness

Notations

- Let Y denote the $n \times p$ matrix containing the data values on p variables for all n units in the sample.
- Let R denote the response indicator as an $n \times p$ 0-1 matrix.
- If y_{ij} is observed, then $r_{ij} = 1$, and if y_{ij} is missing, then $r_{ij} = 0$.
- R is completely observed in the sample.
- For Y, the observed and missing data are denoted as Y_{obs} and Y_{mis}, respectively (i.e. Y_{obs} contain all elements y_{ij} where $r_{ij} = 1$ and Y_{mis} contain all elements y_{ij} where $r_{ij} = 0$).

Types of Missingness

MCAR (Missing Completely at Random)

- Missingness unrelated to observed or unobserved data
- Example: sensor malfunction
- Formal definition:

$$P(R \mid Y_{obs}, Y_{mis}, \psi) = P(R \mid \psi),$$

where ψ is the parameters in the missing data model (i.e. R).

Types of Missingness

MAR (Missing At Random)

- Missingness depends only on observed data
- Example: older patients less likely to report income
- Formal definition:

$$P(R \mid Y_{obs}, Y_{mis}, \psi) = P(R \mid Y_{obs}, \psi)$$

Types of Missingness

MNAR (Missing Not At Random)

- Missingness depends on the value of the missing data
- Example: people with high income refuse to report it
- Formal definition:

$$P(R \mid Y_{obs}, Y_{mis}, \psi) \neq P(R \mid Y_{obs}, \psi)$$

Ignorability

When Can We Ignore the Missingness Mechanism?

- Let θ denote the parameters of scientific interest in the data Y.
- We can ignore the missing data mechanism if:
  - data are MAR or MCAR;
  - parameters θ and ψ are distinct.
- Otherwise (MNAR), need to model the missingness.
- If the nonresponse is ignorable, then the distribution of missing observation does not depend on R. i.e.

$$P(Y_{mis} \mid Y_{obs}, R) = P(Y_{mis} \mid Y_{obs})$$

- Equivalently,

$$P(Y \mid Y_{obs}, R = 1) = P(Y \mid Y_{obs}, R = 0)$$

so the distribution of Y is the same in the response and nonresponse groups.

Ignorability

Consequences of Ignoring Mechanism

- MCAR: complete-case analysis unbiased, but inefficient
- MAR: need to model observed predictors (with exceptions)
- MNAR: bias likely unless missingness model is specified

Single Imputation

- Fills in one value for each missing observation (e.g., mean, regression, hot deck)
- Ignores imputation uncertainty
- Underestimates variance:

$$Var(\hat{\theta})_{SI} < Var(\hat{\theta})_{true}$$

What is Multiple Imputation (MI)?

- A statistical technique for handling missing data
- Imputes missing values multiple times to reflect uncertainty
- More accurate inference than single imputation
- Accounts for uncertainty by simulating from posterior predictive distribution.

Key Idea

- Instead of filling in missing values once, draw from the **posterior predictive distribution** multiple times:

$$Y_{mis} \sim P(Y_{mis} \mid Y_{obs})$$

- Recall when missing is ignorable, $P(Y_{mis} \mid Y_{obs}, R) = P(Y_{mis} \mid Y_{obs})$.
- This distribution is often estimated via regression or machine learning models using observed data.

Ideas behind Multiple Imputation

- A **scientific estimand** Q is a quantity of scientific interest that we can calculate if we would observe the entire population.
- We want to find an **estimate** $\hat{Q}$ that is **unbiased and confidence valid** (Rubin, 1996).
- The possible values of Q given our knowledge of the data Y_{obs} are captured by the posterior distribution $P(Q \mid Y_{obs})$, which can be decomposed into two parts

$$P(Q \mid Y_{obs}) = \int P(Q \mid Y_{obs}, Y_{mis}) P(Y_{mis} \mid Y_{obs}) dY_{mis},$$

where $P(Q \mid Y_{obs}, Y_{mis})$ is the posterior distribution of Q in the hypothetically complete data, and $P(Y_{mis} \mid Y_{obs})$ is the posterior distribution of the missing data given the observed data.

Ideas behind Multiple Imputation – Interpretation

$$P(Q \mid Y_{obs}) = \int P(Q \mid Y_{obs}, Y_{mis}) P(Y_{mis} \mid Y_{obs}) dY_{mis}$$

- Suppose we use $P(Y_{mis} \mid Y_{obs})$ to draw imputations for Y_{mis}, denoted as $\hat{Y}_{mis}$.
- We can then use $P(Q \mid Y_{obs}, \hat{Y}_{mis})$ to calculate the quantity of interest Q from the hypothetically complete data $(Y_{obs}, \hat{Y}_{mis})$.
- We repeat these two steps with new draws $\hat{Y}_{mis}$, and so on.
- The above equation says that the actual posterior distribution of Q is equal to the average over the repeated draws of Q.

Ideas behind Multiple Imputation – Estimate

- It can be shown that the posterior mean of $P(Q \mid Y_{obs})$ is equal to

$$E(Q \mid Y_{obs}) = E(E[Q \mid Y_{obs}, Y_{mis}] \mid Y_{obs}),$$

  i.e. the average of the posterior mean of Q over the repeatedly imputed data.

- This suggests estimate:

$$\bar{Q}_m = \frac{1}{m} \sum_{l=1}^{m} \hat{Q}_l,$$

  where $\hat{Q}_l$ is the estimate of the lth repeated imputation, and m is the number of imputations.

Ideas behind Multiple Imputation – Variation

- The posterior variance of $P(Q \mid Y_{obs})$ is the sum of two variance components:

$$V(Q \mid Y_{obs}) = E[V(Q \mid Y_{obs}, Y_{mis}) \mid Y_{obs}] + V[E(Q \mid Y_{obs}, Y_{mis}) \mid Y_{obs}]$$

- The first component is the average of the repeated complete data posterior variances of Q. This is called the within-variance.
- The second component is the variance between the complete data posterior mean of Q. This is called the between variance.

Ideas behind Multiple Imputation – Variation

- Let $\bar{U}_\infty$ and B_∞ denote the estimated within and between components for an infinitely large number of imputations with $m = \infty$.
- When $m = \infty$, the posterior variance of Q is $T_\infty = \bar{U}_\infty + B_\infty$, where $\bar{U}_m = \frac{1}{m}\sum_{l=1}^{m} \hat{U}_l$, $\hat{U}_l$ is the estimated variance of $\hat{Q}_l$, and $B_m = \frac{1}{m-1}\sum_{l=1}^{m}(\hat{Q}_l - \bar{Q})(\hat{Q}_l - \bar{Q})'$.
- When m is finite, Rubin(1987a, eq. 3.3.5) shows that the posterior variance of Q can be written as:

$$T = \bar{U}_m + B_m + B_m/m = \bar{U}_m + \left(1 + \frac{1}{m}\right) B_m$$

Ideas behind Multiple Imputation – Variation

- The total variance T is from three sources:
  - $\bar{U}$, the variance caused by the fact that we are taking a sample rather than observing the enture population. This is the conventional statistical measure of variability;
  - B_m, the extra variance caused by the fact that there are missing values in the sample;
  - B_m/m, the extra simulation variance caused by the fact that $\bar{Q}$ itself is estimated for finite m.
- The addition of the last term is critical to make multiple imputation work at low values of m. The larger m gets, the smaller the effect of simulation error on the total variance.

3-Steps in Multiple Imputation

1. **Impute**: Each missing value is replaced with a set of m plausible values (typically 5-40+, though modern recommendations lean higher), drawn from a predictive distribution. This creates *m* complete datasets.
   - The imputation model generates these plausible values based on observed relationships in the data.
2. **Analyze**: The intended statistical analysis (e.g., linear regression) is performed independently on each of the *m* completed datasets, yielding *m* sets of parameter estimates (e.g., regression coefficients) and their standard errors.
3. **Pool**: The *m* sets of results are combined into a single set of estimates, standard errors, confidence intervals, and p-values using specific formulas known as Rubin's Rules. This pooling explicitly incorporates the uncertainty due to missing data.

The Imputation Model and The Analysis Model

- **The Imputation Model**: The statistical model(s) used in Step 1 (Impute) to generate imputed values. Its purpose is to predict missing data based on observed data, preserving important relationships and variability. The quality of imputations heavily relies on this model.
- **The Analysis Model** (Substantive Model): The statistical model used in Step 2 (Analyze) to address the researcher's specific scientific questions (e.g., estimating a treatment effect).

Rubin's Rules

- Let Q_m be estimate from the m-th dataset, U_m be variance of Q_m, U_l is the estimated variance of Q_l
- Pooled estimate:

$$\bar{Q} = \frac{1}{m}\sum_{l=1}^{m} Q_l$$

- Within-imputation variance:

$$\bar{U} = \frac{1}{m}\sum_{l=1}^{m} U_l$$

- Between-imputation variance:

$$B = \frac{1}{m-1}\sum_{l=1}^{m}(Q_l - \bar{Q})(Q_l - \bar{Q})'$$

- Total variance:

$$T = \bar{U} + \left(1 + \frac{1}{m}\right) B$$

How to generate multiple imputations

- Predict Method
- Predict + noise method
- Predict + noise + parameter uncertainty
- Additional Predictors
- Drawing from the Observed Data

Imputation Under the Normal Linear Model

Motivation

- Many variables are approximately normally distributed
- Imputation under the normal model assumes:

$$Y = X\beta + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$

Notation

- Y is the variable subject to missing, V is the predictors used to impute Y
- Y_{obs} is the vector containing the n_1 observed data in Y
- Y_{mis} is the vector of n_0 missing data in Y
- $\hat{Y}$ is the imputed values for Y_{mis}
- V_{obs} is the $n_1 \times q$ matrix of predictors of rows for which Y is observed
- V_{mis} is the $n_0 \times q$ matrix of predictors of rows for which Y is missing

Methods

Predict

$$\hat{Y} = \hat{\beta}_0 + V_{mis}\hat{\beta}_1$$

where $\hat{\beta}_0$ and $\hat{\beta}_1$ are least squares estimates calculated from the observed data.

- Note that Y can be either missing covariate or missing outcome in the analysis model. When Y represents a missing covariate, V can include the outcome in the analysis model.
- $\hat{Y}$ is the "best" value in the sense that it is the most likely one under the regression model. However, even the best value may differ from the actual (unknown) value.
- This is "predict" method. The predicted values do not portray this uncertainty, and therefore **cannot** be used as multiple imputations.

R Code

The code to set the parameters and simulate data. We will reuse them in the simulations considered in this section.

```
nsims<-1000 # simulation number
n<-500 # sample size
M<- 10 ## number of imputations
beta0 <- 0; beta1=1; beta2=-1 # parameters to simulate data
beta.hat<-se.betas<-matrix(NA, nrow=nsims, ncol=3) # set the matrix to save results
set.seed(321) # set the random seed to reproduce the results
library(mvtnorm)
# function to simulate MCAR data
simulation <- function(n,beta0,beta1,beta2) {
  v<-rnorm(n)
  x<-rnorm(n,v)
  y<-rnorm(n, beta0+beta1*x+beta2*v,1)
  d.full<-data.frame(y,v,x)
  d.mcar<-d.full
  d.mcar$r<-rbinom(n,1,0.5)
  d.mcar$x<-with(d.mcar, ifelse(r==0, NA, x))
  return(d.mcar)
}
```

Try it yourself – Exercise Day 3A

For the simulated data in the previous slide with MCAR, code the MI with the "Predict" method for the missing X_{mis} using model:

$$\hat{X} = \hat{\alpha}_0 + V\hat{\alpha}_1 + Y\hat{\alpha}_2$$

Remember three steps steps (impute, analysis, pool).

1. How do these regression coefficients and their standard errors compare with those using the full data?

R Code

```
for (j in 1:nsims) {
  d.mcar <- simulation(n,beta0,beta1,beta2) # simulate the data
  # imputation step
  fit.x.mcar<-lm(x ~ y + v, data=d.mcar, subset=(r==1))
  betas.mi<-vars.mi<-matrix(NA, nrow=M, ncol=3)
  for (i in 1:M){
    d.mi.mcar<-d.mcar
    # Predict method, to calculate the linear predictor for missing records
    d.mi.mcar$x[d.mi.mcar$r==0]<- predict(fit.x.mcar, newdata=subset(d.mcar, r==0))
    # analyze step
    mod.mi.mcar<-lm(y ~ x + v, data=d.mi.mcar)
    betas.mi[i,]<-mod.mi.mcar$coeff
    vars.mi[i,]<-diag(vcov(mod.mi.mcar))
  }
  # pool results using Rubin's Rule
  beta.hat[j,]<-colMeans(betas.mi)
  se.betas[j,]<-sqrt(colMeans(vars.mi) + (1+1/M)*c(var(betas.mi[,1]),var(betas.mi[,2]),var(betas.mi[,3])))
}
save(beta0, beta1, beta2, beta.hat, se.betas, file='Pred.RData')
```

```
load(file='Pred.RData')
apply(beta.hat, 2, mean)-c(beta0, beta1, beta2) #bias
```

```
[1]  0.001421192  0.333578438 -0.333876041
```

```
c(mean(beta.hat[,1] - 1.96*se.betas[,1] < beta0 & beta.hat[,1] + 1.96*se.betas[,1]>beta0),
  mean(beta.hat[,2] - 1.96*se.betas[,2]< beta1 & beta.hat[,2] + 1.96*se.betas[,2]> beta1),
  mean(beta.hat[,3] - 1.96*se.betas[,3]< beta2 & beta.hat[,3] + 1.96*se.betas[,3]> beta2)) # coverage probability
```

```
[1] 0.738 0.000 0.003
```

Methods

Predict + noise

$$\hat{Y} = \hat{\beta}_0 + V_{mis}\hat{\beta}_1 + \hat{\epsilon}$$

where $\hat{\epsilon}$ is randomly drawn from the normal distribution as $\hat{\epsilon} \sim N(0, \hat{\sigma}^2)$, and $\hat{\sigma}^2$ is the variance estimate of residual from the observed data.

- This is stochastic regression imputation.

Try it yourself – Exercise Day 3B

For the simulated data in Day 3A, code the MI with the "Predict+noise" method for the missing X_{mis} using model:

$$\hat{X} = \hat{\alpha}_0 + V\hat{\alpha}_1 + Y\hat{\alpha}_2 + \hat{\epsilon}$$

1. Compare these regression coefficients and their standard errors with those using the full data.

R Code

```
for (j in 1:nsims) {
  d.mcar <- simulation(n, beta0, beta1, beta2) # simulate the data
  fit.x.mcar<-lm(x ~ y + v, data=d.mcar, subset=(r==1))
  n0 <- sum(d.mcar$r==0) # number of missing observations
  betas.mi<-vars.mi<-matrix(NA, nrow=M, ncol=3)
  for (i in 1:M){
    d.mi.mcar<-d.mcar
    # update imputation step
    d.mi.mcar$x[d.mi.mcar$r==0]<-
      predict(fit.x.mcar, newdata=subset(d.mcar,r==0)) + rnorm(n0, 0, sd=summary(fit.x.mcar)$sigma)
    mod.mi.mcar<-lm(y ~ x + v, data=d.mi.mcar)
    betas.mi[i,]<-mod.mi.mcar$coeff
    vars.mi[i,]<-diag(vcov(mod.mi.mcar))
  }
  beta.hat[j,]<-colMeans(betas.mi)
  se.betas[j,]<-sqrt(colMeans(vars.mi)+(1+1/M)*c(var(betas.mi[,1]),var(betas.mi[,2]),var(betas.mi[,3])))
}
save(beta0, beta1, beta2, beta.hat, se.betas, file='PredNoise.RData')
```

```
load(file='PredNoise.RData')
apply(beta.hat, 2, mean)-c(beta0, beta1, beta2) #bias
```

```
[1] -0.0002493243  0.0032098713 -0.0023219220
```

```
c(mean(beta.hat[,1] - 1.96*se.betas[,1] < beta0 & beta.hat[,1] + 1.96*se.betas[,1]>beta0),
  mean(beta.hat[,2] - 1.96*se.betas[,2]< beta1 & beta.hat[,2] + 1.96*se.betas[,2]> beta1),
  mean(beta.hat[,3] - 1.96*se.betas[,3]< beta2 & beta.hat[,3] + 1.96*se.betas[,3]> beta2)) # coverage probability
```

```
[1] 0.921 0.924 0.921
```

Methods

Bayesian multiple imputation

$$\dot{X} = \dot{\beta}_0 + V_{mis}\dot{\beta}_1 + \dot{\epsilon}$$

where $\dot{\epsilon} \sim N(0, \dot{\sigma}^2)$, and $\dot{\beta}_0$, $\dot{\beta}_1$, and $\dot{\sigma}$ are random draws from their posterior distribution, given the data.

- This is "predict + noise + parameters uncertainty" method.

Try it yourself – Exercise Day 3C*

For the simulated data in Day 3A, code the MI with the "predict + noise + parameters uncertainty" method for the missing X_{mis} using model:

$$\dot{X} = \dot{\alpha}_0 + V\dot{\alpha}_1 + Y\dot{\alpha}_2 + \dot{\epsilon}$$

where

- $\dot{\alpha} = c(\dot{\alpha}_0, \dot{\alpha}_1, \dot{\alpha}_2) \sim MVN(\hat{\alpha}, \Sigma)$, where $\Sigma = Var(\hat{\alpha})$
- $\dot{\epsilon} \sim N(0, \dot{\sigma}^2)$
- $\dot{\sigma}^2 \sim \chi^2_v \hat{\sigma}^2 / v$, $\hat{\sigma}^2$ is the variance estimate from the observed data, and v is its degree of freedom.

1. Compare these regression coefficients and their standard errors with those using the full data.
2. How do they compare with the IPW estimators that you obtained on Day 2?
3. How do these estimates change if you do not include Y in your imputation model?

R Code

```
for (j in 1:nsims) {
  d.mcar <- simulation(n, beta0, beta1, beta2) # simulate the data
  fit.x.mcar<-lm(x ~ y + v, data=d.mcar, subset=(r==1))
  n0 <- sum(d.mcar$r==0) # number of missing observations
  betas.mi<-vars.mi<-matrix(NA, nrow=M, ncol=3)
  for (i in 1:M){
    d.mi.mcar<-d.mcar
    # update imputation step
    betas.sampled<-rmvnorm(1, fit.x.mcar$coefficients, sigma=vcov(fit.x.mcar))
    sigma2.sampled<-rchisq(1, fit.x.mcar$df.residual)*(summary(fit.x.mcar)$sigma^2)/fit.x.mcar$df.residual
    d.mi.mcar$x[d.mi.mcar$r==0]<- with(subset(d.mi.mcar,r==0), betas.sampled[1,"(Intercept)"]+
            betas.sampled[1,"y"]*y+betas.sampled[1,"v"]*v) + rnorm(n0, 0, sd=sqrt(sigma2.sampled))
    mod.mi.mcar<-lm(y ~ x + v, data=d.mi.mcar)
    betas.mi[i,]<-mod.mi.mcar$coeff
    vars.mi[i,]<-diag(vcov(mod.mi.mcar))
  }
  beta.hat[j,]<-colMeans(betas.mi)
  se.betas[j,]<-sqrt(colMeans(vars.mi) + (1+1/M)*c(var(betas.mi[,1]),var(betas.mi[,2]),var(betas.mi[,3])))
}
save(beta0, beta1, beta2, beta.hat, se.betas, file='BayesianM10.RData')
```

```
load(file='BayesianM10.RData')
apply(beta.hat, 2, mean)-c(beta0, beta1, beta2) #bias
```

```
[1] -0.0013881181  0.0009352306  0.0014239973
```

```
c(mean(beta.hat[,1] - 1.96*se.betas[,1] < beta0 & beta.hat[,1] + 1.96*se.betas[,1]>beta0),
  mean(beta.hat[,2] - 1.96*se.betas[,2]< beta1 & beta.hat[,2] + 1.96*se.betas[,2]> beta1),
  mean(beta.hat[,3] - 1.96*se.betas[,3]< beta2 & beta.hat[,3] + 1.96*se.betas[,3]> beta2)) # coverage probability
```

```
[1] 0.953 0.953 0.948
```

What if we didn't include outcome?

```
for (j in 1:nsims) {
  d.mcar <- simulation(n, beta0, beta1, beta2) # simulate the data
  fit.x.mcar<-lm(x ~ v, data=d.mcar, subset=(r==1))
  n0 <- sum(d.mcar$r==0) # number of missing observations
  betas.mi<-vars.mi<-matrix(NA, nrow=M, ncol=3)
  for (i in 1:M){
    d.mi.mcar<-d.mcar
    # update imputation step
    betas.sampled<-rmvnorm(1, fit.x.mcar$coefficients, sigma=vcov(fit.x.mcar))
    sigma2.sampled<-rchisq(1, fit.x.mcar$df.residual)*(summary(fit.x.mcar)$sigma^2)/fit.x.mcar$df.residual
    d.mi.mcar$x[d.mi.mcar$r==0]<- with(subset(d.mi.mcar,r==0), betas.sampled[1,"(Intercept)"]+
            +betas.sampled[1,"v"]*v) + rnorm(n0, 0, sd=sqrt(sigma2.sampled))
    mod.mi.mcar<-lm(y ~ x + v, data=d.mi.mcar)
    betas.mi[i,]<-mod.mi.mcar$coeff
    vars.mi[i,]<-diag(vcov(mod.mi.mcar))
  }
  beta.hat[j,]<-colMeans(betas.mi)
  se.betas[j,]<-sqrt(colMeans(vars.mi) + (1+1/M)*c(var(betas.mi[,1]),var(betas.mi[,2]),var(betas.mi[,3])))
}
save(beta0, beta1, beta2, beta.hat, se.betas, file='BayesianM1ONY.RData')
```

```
load(file='BayesianM1ONY.RData')
apply(beta.hat, 2, mean)-c(beta0, beta1, beta2) #bias
```

```
[1] -0.0008350917 -0.5004723173  0.5009384474
```

```
c(mean(beta.hat[,1] - 1.96*se.betas[,1] < beta0 & beta.hat[,1] + 1.96*se.betas[,1]>beta0),
  mean(beta.hat[,2] - 1.96*se.betas[,2]< beta1 & beta.hat[,2] + 1.96*se.betas[,2]> beta1),
  mean(beta.hat[,3] - 1.96*se.betas[,3]< beta2 & beta.hat[,3] + 1.96*se.betas[,3]> beta2)) # coverage probability
```

```
[1] 0.976 0.000 0.000
```

Response variable in the imputation model?

It is important to include response variable in the imputation model for missing covariate.

- When covariate is missing, omitting the outcome can bias exposure estimates
- In the above imputation model, need to include outcome y in the V.

Methods

Bootstrap multiple imputation

$$\dot{x} = \dot{\beta}_0 + V_{mis}\dot{\beta}_1 + \dot{\epsilon}$$

where $\dot{\epsilon} \sim N(0, \dot{\sigma}^2)$, and where $\dot{\beta}_0$, $\dot{\beta}_1$, and $\dot{\sigma}$ are the least squares estimates calculated from a bootstrap sample taken from the observed data.

- This is an alternative way to implement "predict + noise + parameters uncertainty" method.

Summary of Part I

- Types of missingness: MCAR, MAR, MNAR
- Ignorability depends on assumptions
- Rubin's rules combine multiple estimates
- Code by Hand for Linear Linear Model
- Next: visualizing missing data and getting started with imputation in R

Overview

Part I: Introduction and Foundations

Part II: Getting Started with Multiple Imputation in R

Part III: Imputation for Different Data Type

Part IV: More Technical Information

Goals of Part II

- Introduce tools for exploring and imputing missing data in R
- Use `mice`, `naniar`, and `VIM` packages

Packages for MI in R

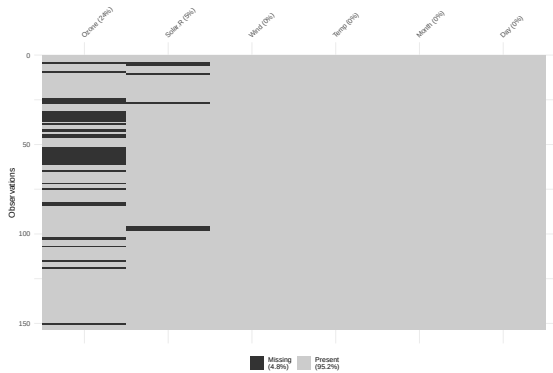
Dataset: `airquality`

```
library(naniar)
library(VIM)
library(mice)
data(airquality)
head(airquality)
```

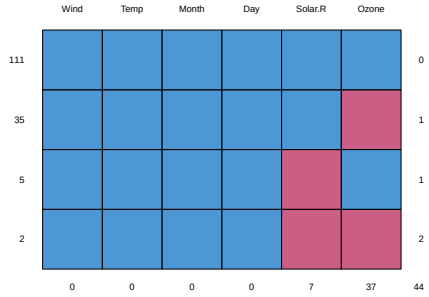
```
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

Missing Data Pattern

```
naniar::vis_miss(airquality)
```



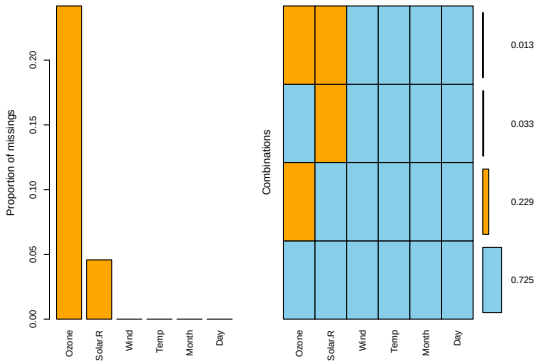

```
mice::md.pattern(airquality)
```



	Wind	Temp	Month	Day	Solar.R	Ozone	
111	1	1	1	1	1	1	0
35	1	1	1	1	1	0	1
5	1	1	1	1	0	1	1
2	1	1	1	1	0	0	2
	0	0	0	0	7	37	44

Aggregated Missingness Map – VIM package

```
AggMap <- aggr(airquality, col=c('skyblue','orange'),
               numbers=TRUE, sortVars=TRUE)
```




```
Variables sorted by number of missings:
Variable       Count
   Ozone 0.24183007
 Solar.R 0.04575163
    Wind 0.00000000
```

```
summary(AggMap)
```

```
Missings per variable:
Variable Count
   Ozone    37
 Solar.R     7
    Wind     0
    Temp     0
   Month     0
     Day     0

Missings in combinations of variables:
Combinations Count   Percent
 0:0:0:0:0:0   111 72.549020
 0:1:0:0:0:0     5  3.267974
 1:0:0:0:0:0    35 22.875817
 1:1:0:0:0:0     2  1.307190
```

```
print(AggMap)
```

```
Missings in variables:
Variable Count
   Ozone    37
 Solar.R     7
```

Summary of Part II

- Visualized missingness with `mice`, `naniar` and `VIM`
- Ready to explore methods for specific types of univariate missing data

Overview

Module 1: Imputation Under the Normal Linear Model

Motivation

- Many variables are approximately normally distributed
- Imputation under the normal model assumes:

$$Y = X\beta + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$

Methods

- `Predict`: in `mice` this method is available as method `norm.predict`. **Not Recommended**
- `Predict + noise`: in `mice` this method is available as method `norm.nob`. **Not Recommended**
- `Bayesian multiple imputation`: in `mice` this method is available as method `norm`. **Recommended**
- `Bootstrap multiple imputation`: in `mice` this method is available as method `norm.boot`. **Recommended**

Run Imputation with Normal Linear Model

```
imp_air_norm <- mice(airquality, m = 5, method = "norm",
                     seed = 2025)
```

```
 iter imp variable
  1   1  Ozone  Solar.R
  1   2  Ozone  Solar.R
  1   3  Ozone  Solar.R
  1   4  Ozone  Solar.R
  1   5  Ozone  Solar.R
  2   1  Ozone  Solar.R
  2   2  Ozone  Solar.R
  2   3  Ozone  Solar.R
  2   4  Ozone  Solar.R
  2   5  Ozone  Solar.R
  3   1  Ozone  Solar.R
  3   2  Ozone  Solar.R
  3   3  Ozone  Solar.R
  3   4  Ozone  Solar.R
  3   5  Ozone  Solar.R
  4   1  Ozone  Solar.R
  4   2  Ozone  Solar.R
  4   3  Ozone  Solar.R
  4   4  Ozone  Solar.R
  4   5  Ozone  Solar.R
  5   1  Ozone  Solar.R
  5   2  Ozone  Solar.R
  5   3  Ozone  Solar.R
  5   4  Ozone  Solar.R
  5   5  Ozone  Solar.R
```

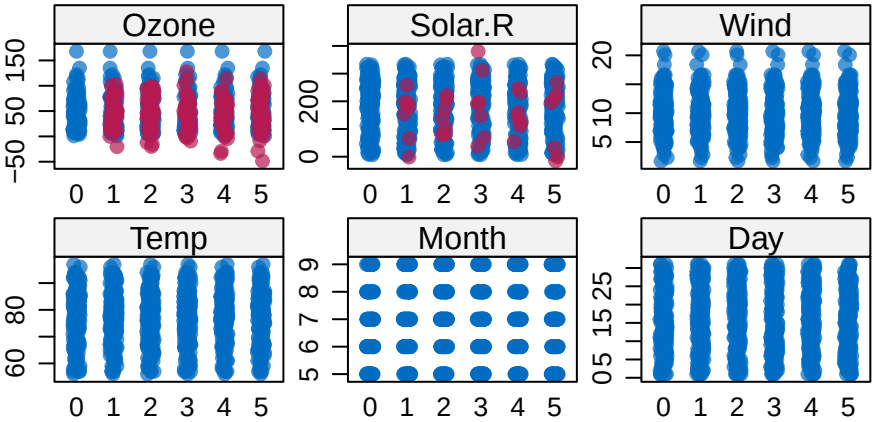
```
summary(imp_air_norm)
```

```
Class: mids
Number of multiple imputations:  5
Imputation methods:
  Ozone Solar.R    Wind    Temp   Month     Day
 "norm"  "norm"      ""      ""      ""      ""
PredictorMatrix:
        Ozone Solar.R Wind Temp Month Day
Ozone       0       1    1    1     1   1
Solar.R     1       0    1    1     1   1
Wind        1       1    0    1     1   1
Temp        1       1    1    0     1   1
Month       1       1    1    1     0   1
Day         1       1    1    1     1   0
```

Visualizing the Imputations

Diagnostics: Stripplot

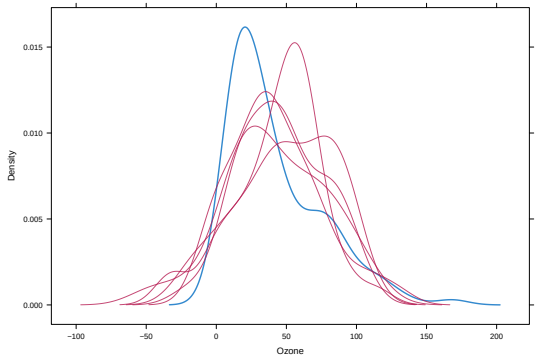
```
stripplot(imp_air_norm, pch = 20, cex = 1.2)
```



Visualizing the Imputations

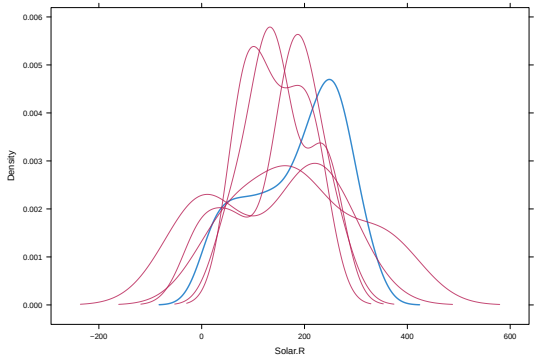
Diagnostics: Density Plot for Ozone

```
densityplot(imp_air_norm, ~Ozone)
```



Diagnostics: Density Plot for Solar.R

```
densityplot(imp_air_norm, ~Solar.R)
```



Review the Imputed Values

```
complete_air_long <- complete(imp_air_norm, action = "long",
                              include = TRUE)
head(complete_air_long)
```

```
  Ozone Solar.R Wind Temp Month Day .imp .id
1    41     190  7.4   67     5   1    0   1
2    36     118  8.0   72     5   2    0   2
3    12     149 12.6   74     5   3    0   3
4    18     313 11.5   62     5   4    0   4
5    NA      NA 14.3   56     5   5    0   5
6    28      NA 14.9   66     5   6    0   6
```

```
tail(complete_air_long)
```

```
       Ozone Solar.R Wind Temp Month Day .imp .id
913 14.00000      20 16.6   63     9  25    5 148
914 30.00000     193  6.9   70     9  26    5 149
915 17.51653     145 13.2   77     9  27    5 150
916 14.00000     191 14.3   75     9  28    5 151
917 18.00000     131  8.0   76     9  29    5 152
918 20.00000     223 11.5   68     9  30    5 153
```

Model Fitting on Each Imputed Dataset

Obtain the fitted model for each imputed dataset

```
fit_air_norm <- with(imp_air_norm, lm(Ozone ~ Temp + Wind))
summary(fit_air_norm)
```

```
# A tibble: 15 x 7
   term        estimate std.error statistic  p.value  nobs df.residual
   <chr>          <dbl>     <dbl>     <dbl>    <dbl> <int>       <dbl>
 1 (Intercept)   -53.9     19.5       -2.77 6.32e- 3   153         150
 2 Temp            1.69     0.208      8.11 1.66e-13   153         150
 3 Wind           -3.45     0.559     -6.17 6.17e- 9   153         150
 4 (Intercept)   -77.4     20.8       -3.72 2.83e- 4   153         150
 5 Temp            1.92     0.222      8.61 9.37e-15   153         150
 6 Wind           -2.77     0.598     -4.64 7.69e- 6   153         150
 7 (Intercept)   -58.2     20.5       -2.83 5.23e- 3   153         150
 8 Temp            1.70     0.219      7.77 1.17e-12   153         150
 9 Wind           -3.25     0.589     -5.51 1.52e- 7   153         150
10 (Intercept)   -78.2     19.4       -4.04 8.44e- 5   153         150
11 Temp            1.92     0.207      9.29 1.72e-16   153         150
12 Wind           -2.89     0.556     -5.20 6.42e- 7   153         150
13 (Intercept)   -66.4     20.1       -3.30 1.23e- 3   153         150
14 Temp            1.84     0.215      8.54 1.37e-14   153         150
15 Wind           -3.48     0.578     -6.02 1.27e- 8   153         150
```

Pooled Estimates Using Rubin's Rules

```
summary(pool(fit_air_norm))
```

```
         term   estimate  std.error statistic       df       p.value
1 (Intercept) -66.823862 23.4078306 -2.854765 37.43732 6.983309e-03
2        Temp   1.812822  0.2470602  7.337573 41.37853 5.235268e-09
3        Wind  -3.167079  0.6762146 -4.683541 35.61243 4.018597e-05
```

Exercise Day 3D

Fit a multiple imputation estimator using the mice package with Bayesian MI approach in R for the data generated in Day 3A.

1. How do these regression coefficients and their standard errors compare with those that you calculated by hand using Bayesian MI approach (Day 3C)?

Summary of Module 1

- Imputation under a normal linear model assumes Gaussian residuals
- Posterior draws used to impute missing values

Module 2: Imputation Under Non-Normal Distributions

Why Normal Models Can Fail

- Real-world data are often:
  - Skewed (e.g., income, biomarkers)
  - Bounded (e.g., rates, percentages)
  - Heavy-tailed (e.g., medical costs)

Problem with Normal Model

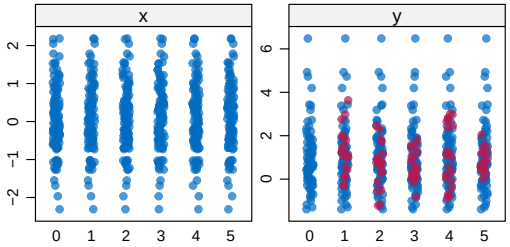
- Demirtas, Freels, and Yucel (2008) found that flatness of the density, heavy tails, non-zero peakedness, skewness and multimodality do not appear to hamper the good performance of multiple imputation for the mean structure in samples $n > 400$, even for high percentages (75%) of missing data in one variable.
- The variance parameter is more critical though, and could be off-target in smaller samples.

Alternative Approaches

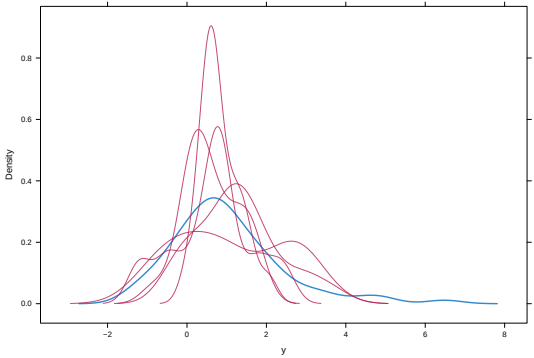
1. **Transformations**
   - Log, square root, Box-Cox
2. **Nonparametric methods**
   - Predictive Mean Matching (PMM) – next module
3. **Robust methods**
   - The `ImputeRobust` package implements various `mice` methods for continuous data: method=`gamlss` (normal), method=`gamlssJSU` (Johnson's SU), method=`gamlssTF` (t-distribution) and method=`gamlssGA` (gamma distribution).

Simulated Non-Normal Example

```
library("ImputeRobust")
library("gamlss")
set.seed(123)
n <- 100
x <- rnorm(n)
y <- rt(n, df=10, ncp=1+0.5*x)
y[sample(1:n, 20)] <- NA
non_normal <- data.frame(x, y)
imp_non_normal <- mice(non_normal, m = 5, method = "gamlssTF",
            seed = 88009, print = FALSE)
stripplot(imp_non_normal, pch = 20, cex = 1.2)
```




```
densityplot(imp_non_normal, ~y)
```



Summary of Module 2

- Normal imputation may perform poorly on skewed data
- `ImputeRobust` together with `gamlss` package provides a series of additional imputation models
- Next module: PMM offers a robust alternative

Module 3: Predictive Mean Matching (PMM)

What is Predictive Mean Matching?

- A semi-parametric method
- For each missing value:
  - Predict outcome using regression
- Find observed values with closest predicted means (hot deck)
- Randomly draw one from them as the donor

Why Use PMM?

- Preserves original data distribution
- Ensures plausible imputed values
- Works well for:
  - Non-normal data
- Skewed or bounded distributions
- Small samples

PMM in Equation Form

Let $\hat{y}_{mis}$ be the predicted value for missing observation. Let $\hat{y}_{obs}$ be predictions for complete cases.

There are various ways to perform PMM.

- Take the closest candidate. For each case j from y_{mis}, take the case i from y_{obs} such that $|\hat{y}_i - \hat{y}_j|$ is minimal as the donor and take $y_{obs,i}$ (the observed value) as imputed value for $y_{mis,i}$. This is "deterministic hot deck".
- Choose a threshold η, for each case j from y_{mis}, take all i from y_{obs} for which $|\hat{y}_i - \hat{y}_j| < \eta$ as candidate donors for imputing case j. Randomly sample one donor from the candidates, and take its observed value as the imputed value for $y_{mis,i}$.
- Find the d candidates for which $|\hat{y}_i - \hat{y}_j|$ is minimal, and sample one of them. Usual values for d are 3, 5 and 10 (default is 5 in `mice`, can be changed using `donors` option).
- Sample one donor with a probability that depends on $|\hat{y}_i - \hat{y}_j|$ (Siddique and Belin 2008).

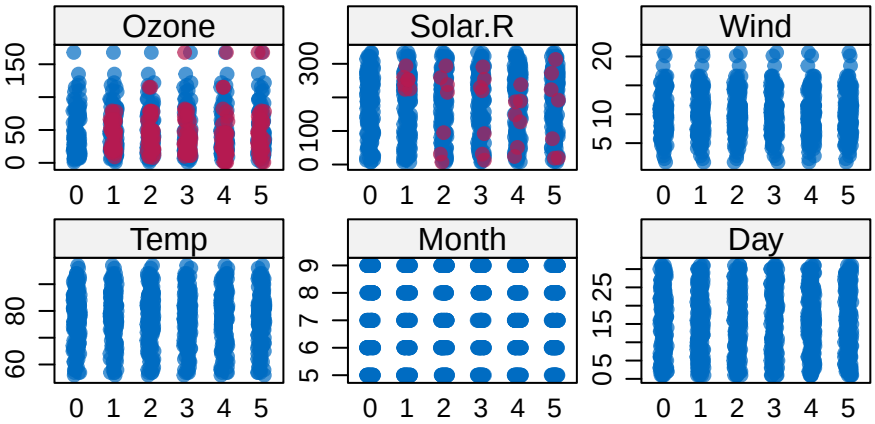
Run PMM Imputation in R

```
imp_air_pmm <- mice(airquality, method = "pmm", m = 5, donors=5, seed = 2025)
```

```
 iter imp variable
  1   1  Ozone  Solar.R
  1   2  Ozone  Solar.R
  1   3  Ozone  Solar.R
  1   4  Ozone  Solar.R
  1   5  Ozone  Solar.R
  2   1  Ozone  Solar.R
  2   2  Ozone  Solar.R
  2   3  Ozone  Solar.R
  2   4  Ozone  Solar.R
  2   5  Ozone  Solar.R
  3   1  Ozone  Solar.R
  3   2  Ozone  Solar.R
  3   3  Ozone  Solar.R
  3   4  Ozone  Solar.R
  3   5  Ozone  Solar.R
  4   1  Ozone  Solar.R
  4   2  Ozone  Solar.R
  4   3  Ozone  Solar.R
  4   4  Ozone  Solar.R
  4   5  Ozone  Solar.R
  5   1  Ozone  Solar.R
  5   2  Ozone  Solar.R
  5   3  Ozone  Solar.R
  5   4  Ozone  Solar.R
  5   5  Ozone  Solar.R
```

Visualizing PMM Imputation

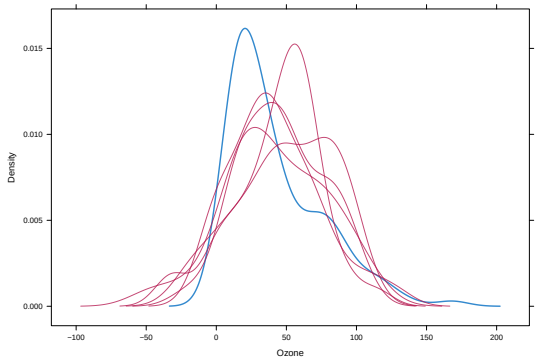
```
stripplot(imp_air_pmm, pch = 20, cex = 1.2)
```



Consider PMM as Robust Alternative for normal data

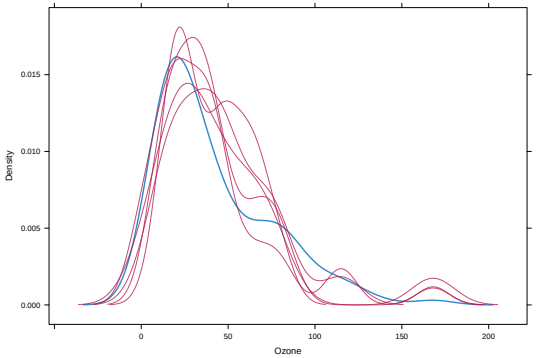
method=‘norm’

```
densityplot(imp_air_norm, ~Ozone)
```



method=‘pmm’

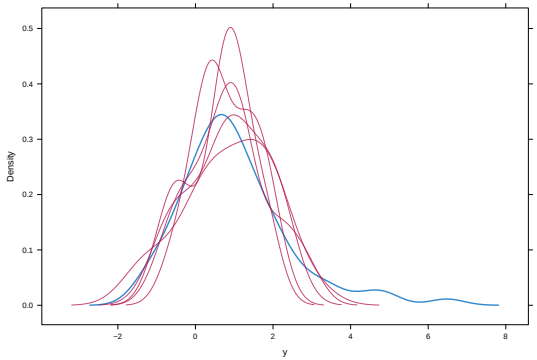
```
densityplot(imp_air_pmm, ~Ozone)
```



Consider PMM as Robust Alternative for non-normal data

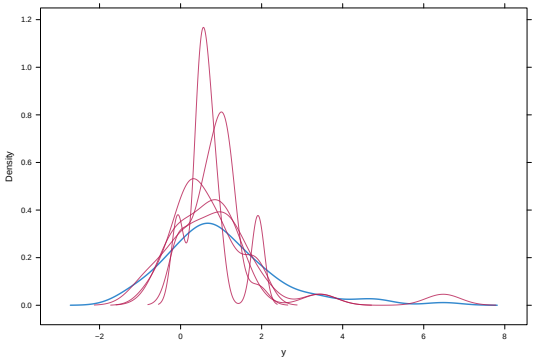
method="gamlssTF"

```
imp_non_normal <- mice(non_normal, method = "gamlssTF",
                       m = 5, printFlag = F)
densityplot(imp_non_normal, ~y)
```



method="pmm"

```
imp_non_normal_pmm <- mice(non_normal, method = "pmm",
                           m = 5, printFlag = F)
densityplot(imp_non_normal_pmm, ~y)
```



Fit Model and Pool Results

```
fit_air_pmm <- with(imp_air_pmm, lm(Ozone ~ Temp + Wind))
summary(pool(fit_air_pmm))
```

```
         term   estimate  std.error statistic       df      p.value
1 (Intercept) -68.753948 27.6713870 -2.484659 14.02943 2.620148e-02
2        Temp   1.804572  0.2948024  6.121294 14.18639 2.496363e-05
3        Wind  -2.975598  0.6733450 -4.419128 34.50843 9.359592e-05
```

Exercise Day 3E

For the data generated in Day 3A, use PMM method and compare the results from Exercise Day 3D.

Summary of Module 3

- PMM is robust and widely used
- Produces realistic values drawn from observed donors
- Useful when model assumptions are questionable

Module 4: Categorical Data Imputation

Types of Categorical Variables

- **Binary** (0/1, yes/no): logistic regression, where the outcome probability is modeled as

$$Pr(y_i = 1 \mid X_i, \beta) = \frac{\exp(X_i\beta)}{1 + \exp(X_i\beta)}$$

- `"logreg"`: logistic regression (binary)

Types of Categorical Variables

- **Nominal** (unordered categories): multinomial logit regression for K unordered categories,

$$Pr(y_i = k \mid X_i, \beta) = \frac{\exp(X_i \beta_k)}{\sum_{k=1}^{K} \exp(X_i \beta_k)}$$

  for $k = 1, ..., K$, where β_k varies over the categories and where $\beta_1 = 0$ to identify the model.
- `"polyreg"`: multinomial logistic regression (nominal)

Types of Categorical Variables

- **Ordinal** (ordered categories): ordered logit model or proportional odds model

$$Pr(y_i \leq k \mid X_i, \beta, \tau_k) = \frac{\exp(\tau_k - X_i\beta)}{1 + \exp(\tau_k - X_i\beta)}$$

with the slope β is identical across categories, but the intercepts τ_k differ. For identification, we set $\tau_1 = 0$. The probability of observing category k is written as

$$Pr(y_i = k \mid X_i) = Pr(y_i \leq k \mid X_i) - Pr(y_i \leq k - 1 \mid X_i)$$

where the model parameters β, τ_k and τ_{k-1} are suppressed for clarity.

- `"polr"`: proportional odds model (ordinal).

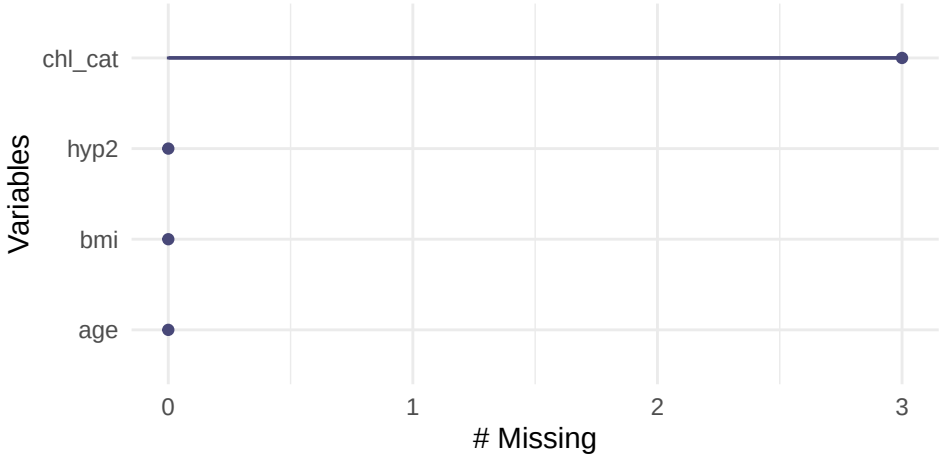
Dataset: nhanes (in mice package)

```r
library(mice)
library(dplyr)
data(nhanes)
nhanes$chl_cat <- cut(
  nhanes$chl,
  breaks = c(-Inf, 199, 239, Inf),
  labels = c('Low', 'Normal', 'High'), # 1-Desirable <200, 2-Borderline High, [200-239], 3-High if >=240
  ordered=FALSE,
  right = TRUE
)
nhanes3 <- nhanes %>%
  mutate(hyp2=factor(hyp)) %>%
  select(c('age', 'bmi', 'hyp2','chl_cat')) %>%
  filter(!is.na(hyp2) & !is.na(bmi))
summary(nhanes3)
```

```
      age             bmi        hyp2     chl_cat
 Min.   :1.000   Min.   :20.40   1:12   Low   :7
 1st Qu.:1.000   1st Qu.:22.65   2: 4   Normal:5
 Median :2.000   Median :26.75          High  :1
 Mean   :1.812   Mean   :26.56          NA's  :3
 3rd Qu.:2.250   3rd Qu.:28.93
 Max.   :3.000   Max.   :35.30
```

Visualizing Categorical Missingness

```
library(naniar)
gg_miss_var(nhanes3)
```



Imputing with proportional odds model

```
imp_cat <- mice(nhanes3, method = c("", "", "", "polyreg"),
                m = 5, seed = 2025, printFlag = F)
summary(imp_cat)
```

```
Class: mids
Number of multiple imputations:  5
Imputation methods:
      age        bmi       hyp2   chl_cat
       ""         ""         "" "polyreg"
PredictorMatrix:
        age bmi hyp2 chl_cat
age       0   1    1       1
bmi       1   0    1       1
hyp2      1   1    0       1
chl_cat   1   1    1       0
```

Review the imputed data

```
table(nhanes3$chl_cat, useNA = 'ifany')
```

```
   Low Normal   High   <NA>
     7      5      1      3
```

```
table(complete(imp_cat, 1)$chl_cat)
```

```
   Low Normal   High
     7      7      2
```

```
table(complete(imp_cat, 2)$chl_cat)
```

```
   Low Normal   High
     8      6      2
```

```
table(complete(imp_cat, 3)$chl_cat)
```

```
   Low Normal   High
     7      6      3
```

```
table(complete(imp_cat, 4)$chl_cat)
```

```
   Low Normal   High
     8      5      3
```

```
table(complete(imp_cat, 5)$chl_cat)
```

```
   Low Normal   High
     8      6      2
```

Model Fit and Pooling

```
fit_cat <- with(imp_cat, glm(hyp2 ~ chl_cat, family = binomial))
summary(pool(fit_cat))
```

```
           term   estimate   std.error    statistic        df   p.value
1   (Intercept) -1.8842499    1.077465 -1.748779866 11.289452 0.1074325
2 chl_catNormal  0.4662345    1.596358  0.292061284  9.808275 0.7763239
3   chl_catHigh  9.5879362 2917.033213  0.003286879 11.373862 0.9974344
```

Notes

- The methods are based on the generalized linear models. However, the methods may be unstable, slow and exhibit poor performance.
- Audigier, Husson, and Josse (2017) found that logistic regression presented difficulties on the datasets with a high number of categories, resulting in undercoverage on several quantities.
- In many datasets, especially those with many categories, the ratio of the number of fitted parameters relative to the number of events easily drops below 10, which may lead to estimation problems. In those cases, the advice is to specify more robust methods, like `pmm`, `cart` (classification and regression trees) or `rf` (random forest imputation).

Summary of Module 4

- Use logistic models for binary variables
- Use multinomial models for nominal data
- Visualize before and after imputation to check plausibility

Module 5: Other Data Types

Count Data

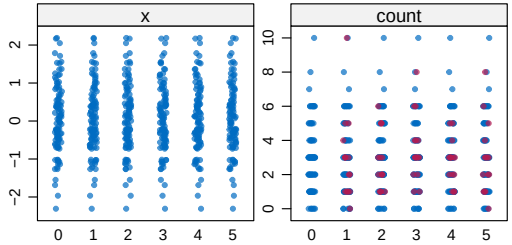
- Examples: number of hospital visits, events
- Count data can be imputed in various ways:
  - Predictive mean matching (cf. Section 3.4).
  - Ordered categorical imputation (cf. Section 3.6).
  - (Zero-inflated) Poisson regression (Raghunathan et al. 2001).
  - (Zero-inflated) negative binomial regression (Royston 2009).
- `ImputeRobust` package implements the following mice methods for count data: gamlssPO (Poisson), gamlssZIBI (zero-inflated binomial) and gamlssZIP (zero-inflated Poisson).

Simulated Count Data

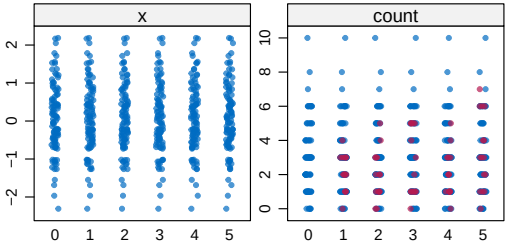
```
set.seed(123)
sim_count <- data.frame(
  x = rnorm(100),
  count = rpois(100, lambda = 3+x)
)
sim_count$count[sample(1:100, 15)] <- NA
```

Impute Using PMM

```
imp_count <- mice(sim_count, method = "pmm", m = 5, printFlag = F)
stripplot(imp_count, pch = 20)
```




```
imp_count_PO <- mice(sim_count, method = "gamlssPO", m = 5, printFlag
stripplot(imp_count_PO, pch = 20)
```



Semi-continuous Data

- Examples: cost, alcohol use
- Often has many zeros and positive skew
- Consider PMM with transformation or two-part models
  - Two parts: The first step is to determine whether the imputed value is zero or not. The second step is only done for those with a non-zero value, and consists of drawing a value from the continuous part.
  - Implementation: `mi` (Su et al. 2011) and `irmi` (Templ, Kowarik, and Filzmoser 2011).

Censored, Truncated and Rounded Data

- Censored: lab value below detection limit
- Rounded: values rounded to nearest unit (e.g., age)
  - Requires models that incorporate bounds
  - Use packages like `survival`, `MIICD` package
- Truncated: values below/above thresholds not observed
  - Use packages like `truncreg`

Imputation Approaches

Data Type	Suggested Method
Count	PMM, transform
Semi-continuous	Two-part model, PMM
Censored	Tobit, survival models
Truncated	Truncated regression models
Rounded	Custom, rounding models

Summary of Module 5

- Imputation strategy depends on scale and distribution
- PMM is flexible when specialized methods are unavailable
- Advanced cases may need custom modeling

Module 6: Classification and Regression Trees (CART)

What is CART?

- Classification and regression trees (CART) (Breiman et al. 1984) are a popular class of machine learning algorithms.
- The target variable can be discrete (classification tree) or continuous (regression tree).
- Nonparametric imputation method
- Captures: Nonlinear relationships and interactions
- Useful when:
  - Parametric assumptions don't hold
  - Many categorical predictors
- Splits data recursively on best predictor
- Each missing value imputed from terminal node

Implication

- Packages that fail to incorporate uncertainty:
  - The `missForest` package successfully used regression and classification trees to predict the outcomes in mixed continuous/categorical data. It is popular, however, it does not account for the uncertainty caused by the missing data.
- Packages that incorporate uncertainty:
  - `CALIBERrfimpute` package with methods `"rfcat"` and `"rfcont"`
  - Doove, Van Buuren, and Dusseldorp (2014) independently developed a similar set of routines building on the `rpart` and `randomForest` packages.
  - `mice` package using methods `"cart"` and `"rf"`.

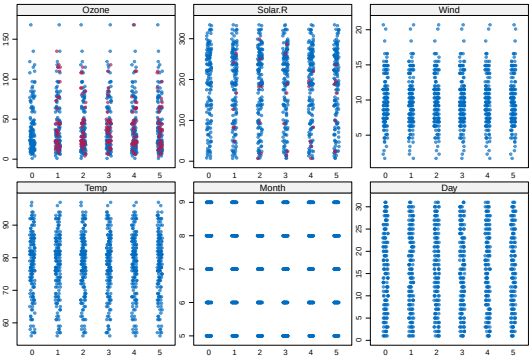
CART in mice()

```
imp_cart <- mice(airquality, method = "cart", m = 5, seed = 2025)
```

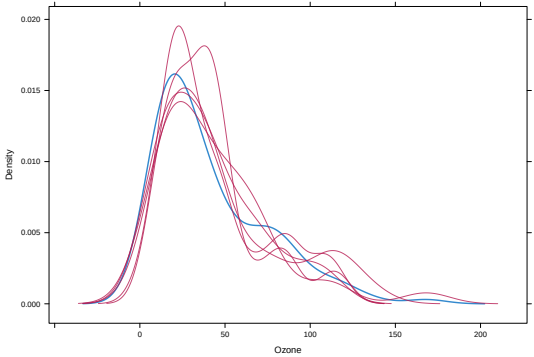
```
 iter imp variable
  1   1  Ozone  Solar.R
  1   2  Ozone  Solar.R
  1   3  Ozone  Solar.R
  1   4  Ozone  Solar.R
  1   5  Ozone  Solar.R
  2   1  Ozone  Solar.R
  2   2  Ozone  Solar.R
  2   3  Ozone  Solar.R
  2   4  Ozone  Solar.R
  2   5  Ozone  Solar.R
  3   1  Ozone  Solar.R
  3   2  Ozone  Solar.R
  3   3  Ozone  Solar.R
  3   4  Ozone  Solar.R
  3   5  Ozone  Solar.R
  4   1  Ozone  Solar.R
  4   2  Ozone  Solar.R
  4   3  Ozone  Solar.R
  4   4  Ozone  Solar.R
  4   5  Ozone  Solar.R
  5   1  Ozone  Solar.R
  5   2  Ozone  Solar.R
  5   3  Ozone  Solar.R
  5   4  Ozone  Solar.R
  5   5  Ozone  Solar.R
```

Visualizing CART Imputations

```
stripplot(imp_cart, pch = 20)
```




```
densityplot(imp_cart, ~Ozone)
```



Fit Model and Pool Results

```
fit_cart <- with(imp_cart, lm(Ozone ~ Temp + Wind))
summary(pool(fit_cart))
```

```
         term   estimate  std.error statistic        df      p.value
1 (Intercept) -77.702538 21.4632698 -3.620256  90.42277 4.849851e-04
2        Temp   1.884259  0.2219442  8.489788 125.30207 5.028512e-14
3        Wind  -2.684573  0.6444373 -4.165762  56.64063 1.070719e-04
```

Advantages of CART

- Handles nonlinearities and interactions
- Model-free
- Often more robust than linear model imputation

Summary of Module 6

- CART is a flexible, tree-based imputation method
- Available in several packages including `mice(method = "cart")` and `mice(method = "rf")`
- Works well with mixed data types and complex relationships

Overview

Part I: Introduction and Foundations

Part II: Getting Started with Multiple Imputation in R

Part III: Imputation for Different Data Type

Part IV: More Technical Information

Scope of the Imputation Model

- **Broad**. Create one set of imputations to be used for all projects and analyses. A broad scope is appropriate for publicly released data, cohort data and registers, where different people use the data for different purposes.
- **Intermediate**. Create one set of imputations per project and use this set for all analyses. An intermediate scope is appropriate for analyses that estimate relatively similar quantities. The imputer and analyst can be different persons.
- **Narrow**. A separate imputed dataset is created for each analysis. The imputer and analyst are typically the same person. A narrow scope is appropriate if the imputed data are used only to estimate the same quantity. Different analyses require different imputations.

Congeniality: Models in Agreement

- **Distinct Models, Interdependent Validity**: The imputation and analysis models are distinct and may even be specified by different individuals (e.g., a data provider imputes, an analyst later uses the data). However, the validity of the final pooled results from the analysis model critically depends on the relationship between these two models. If an analyst's model makes different assumptions than the imputer's (e.g., includes an interaction term the imputer didn't consider), the statistical foundation for Rubin's Rules is weakened, potentially leading to invalid inferences. This underscores the need for clear documentation of the imputation model.
- **Definition**: Congeniality exists when the imputation model and the analysis model are "in agreement" or "compatible." They make consistent assumptions about the data's underlying structure, including relationships between variables, types of effects (linear, non-linear), and relevant interactions.
  - More formally, Meng (1994) defined two procedures as congenial if they can be derived from a single, overarching Bayesian model, implying the imputation model correctly reflects all aspects of the data that the analysis model will later examine.

Congeniality: Models in Agreement

- **Why Congeniality is Non-Negotiable**: It is a fundamental requirement for Rubin's Rules to yield statistically valid inferences-unbiased parameter estimates, correct standard errors, accurate confidence intervals, and reliable p-values. Without congeniality, tests may not have the correct size, and confidence intervals may not achieve their nominal coverage.
- In settings with non-congeniality, Rubin's Rules for computing the variance may lead to poor coverage, but that these can often be corrected by bootstrapping. Bootstrapping + MI is recommended (as opposed to MI + bootstrapping). This can be computationally intensive.

Variance Ratios

- **Proportion of the variation attributable to the missing data**

$$\lambda = \frac{B + B/m}{T}$$

- **Relative increase in variance due to nonresponse** (Rubin 1987b eq. 3.1.7)

$$r = \frac{B + B/m}{\bar{U}}$$

$r = \lambda/(1 - \lambda)$.

- **Fraction of information about Q missing due to nonresponse** (Rubin 1987b eq. 3.1.10)

$$\gamma = \frac{r + 2/(v + 3)}{1 + r}$$

where v is the degree of freedom (next slide).

Degrees of Freedom

The degrees of freedom is the number of observations after accounting for the number of parameters in the model.

- The old formula (Rubin 1987b eq. 3.1.6):

$$v_{old} = (m-1)(1+1/r^2) = \frac{m-1}{\lambda^2}$$

  where λ is the proportion of the variation attributable to the missing data, and r is the relative increase in variance due to missingness, and $r = \lambda/(1-\lambda)$.

- The lowest possible value is $v_{old} = m-1$, which occurs if essentially all variation is attributable to the non-response. The highest value $v_{old} = \infty$ indicates that all variation is sampling variation, either because there were no missing data, or because we could re-create them perfectly.

- Problem: v_{old} can produce values that are larger than the sample size in the complete data, a situation that is clearly inappropriate.

Degrees of Freedom

- **Revise it**:
  - The estimated observed data degrees of freedom that accounts for the missing information is

$$v_{obs} = \frac{(n-k+1)}{(n-k+3)}(n-k)(1-\lambda)$$

    where n is the sample size for the observed data, and k is the number of parameters.
  - The adjusted degrees of freedom is:

$$v = \frac{v_{old}\, v_{obs}}{v_{old} + v_{obs}}$$

  - Note $v \leq v_{com}$

Statistical Intervals and Tests

When Q is a scalar, or we test each of the parameters

- Since the total variance of T is not known a priori, $\bar{Q}$ follows a t-distribution rather than the normal. Univariate tests are based on the approximation

$$\frac{Q - \bar{Q}}{T} \sim t_v$$

where t_v is the Student's t-distribution with v degrees of freedom, with v defined by Equation (2.32).

- The $100(1-\alpha)\%$ confidence interval of a $\bar{Q}$ is calculated as

$$\bar{Q} \pm t_{v,1-\alpha/2}\sqrt{T}$$

where $t_{v,1-\alpha/2}$ is the quantile corresponding to probability $1-\alpha/2$ of t_v.

- $H_0 : Q = Q_0$ for some fixed value Q_0. We can find the p-value of the test as:

$$P_s = Pr\left[F_{1,v} > \frac{(Q_0 - \bar{Q})^2}{T}\right],$$

where $F_{1,v}$ is an F-distribution with 1 and v degrees of freedom.

Summary of Notation

Quantity	Symbol(s)	Formula	Interpretation
Pooled Estimate	$\bar{Q}$	$\frac{1}{m}\sum \widehat{Q}_i$	Average estimate across imputations
Within-Imputation Variance	$\bar{U}$	$\frac{1}{m}\sum \widehat{U}_i$	Average sampling variance within imputations
Between-Imputation Variance	B	$\frac{1}{m-1}\sum (\widehat{Q}_i - \bar{Q})^2$	Variance across estimates due to missing data
Total Variance	T	$\bar{U} + \left(1 + \frac{1}{m}\right) B$	Total uncertainty (sampling + missing data)
Pooled Standard Error	SE_{pooled}	$\sqrt{T}$	Standard error for the pooled estimate
Relative Increase Variance	r	$\frac{(1+1/m)B}{\bar{U}}$	Proportional increase in variance due to missingness
Degrees of Freedom	v	$(m-1)(1+1/r)^2$ (Basic)	Adjusted df for t-distribution inference
Fraction Missing Info (FMI)	γ	$\approx \frac{r+2/(v+3)}{1+v}$ (Approx.)	Proportion of variance due to missing data