Missing Data

Randy Johnson

Types of missing data

- Missing completely at random
- Missing at random
- Informative missing

Types of missing data

Missing completely at random: no factors relating to the samples (measured or not) influenced which data are missing.

- Example: A study ends prematurely, and some data are not able to be collected, independent of any sample characteristics.

Types of missing data

Missing at random: some important factors may have influenced which data are missing, but the probability of missingness is a function of variables that were measured.

- Example: Individuals with depression may be more likely to be lost to followup, resulting in missing data. As long as loss to followup isn't related to the variable that is missing (e.g. number of cigarettes smoked each week during the study), we can assume that the number of cigarettes smoked by individuals we did observe is representative of the number of cigarettes smoked by individuals we didn't observe.

Types of missing data

Informative missing: missing data are biased in some way by other confounding variables. This results in estimates that are higher or lower than they should be. This is often nearly impossible to detect without some outside information (e.g. experience with past studies or knowledge of the population under study).

- Example: Some unmeasured confounding variable (location: poor neighborhood) influences heavy smokers in the treatment group to drop out of the study at a higher rate than individuals who smoke less.

Problems stemming from missing data

- What is the rate of missingness in each group?
- Are there any factors in our disease model that would cause an individual to have missing data?
- What other relationships between observed data and missingness exist?

Informative missingness can cause unexpected problems, including false associations.

Dealing with missing data: Ignore missing data

```
require(missForest)
set.seed(239847)
n <- 100
# generate a dataset with a lot of missing data
mcar <- data_frame(x1 = rnorm(n),
                   x2 = rnorm(n),
                   g = rbinom(n, 1, .5),
                   y = x1 + x2 + x1*x2 + (g == 1) + rnorm(n)) %>%
       prodNA(noNA = 0.1)
summary(mcar)
```

```
##        x1                  x2                   g
##  Min.   :-2.60883   Min.   :-2.867363   Min.   :0.0000
##  1st Qu.:-0.72759   1st Qu.:-0.655759   1st Qu.:0.0000
##  Median :-0.08719   Median : 0.001891   Median :0.0000
##  Mean   :-0.08410   Mean   : 0.020576   Mean   :0.4783
##  3rd Qu.: 0.58334   3rd Qu.: 0.706346   3rd Qu.:1.0000
##  Max.   : 1.97368   Max.   : 2.190697   Max.   :1.0000
##  NA's   :10         NA's   :15          NA's   :8
##        y
##  Min.   :-6.35206
##  1st Qu.:-0.56756
##  Median : 0.08151
##  Mean   : 0.41975
##  3rd Qu.: 1.11505
##  Max.   : 6.14622
##  NA's   :7
```

Dealing with missing data: Ignore missing data

```r
# look at the relationship between x and y by g
require(gmodels)
```

```
## Loading required package: gmodels
```

```r
(lm(y ~ x1*x2 + g, data = mcar) %>%
   ci())[,1:3]
```

```
##                Estimate   CI lower  CI upper
## (Intercept) 0.04631757 -0.3090120 0.4016471
## x1          0.90845393  0.6465163 1.1703915
## x2          0.90704660  0.6286759 1.1854173
## g           0.70943829  0.1786526 1.2402240
## x1:x2       1.48408561  1.1797384 1.7884329
```

Dealing with missing data: Replace missing data with the group mean

```r
head(mcar)
```

```
## # A tibble: 6 x 4
##        x1     x2     g      y
##     <dbl>  <dbl> <int>  <dbl>
## 1  0.438  2.03      NA  3.43
## 2  0.512 -0.432      1 -0.320
## 3  0.374  0.214      0  1.02
## 4  0.594  1.01       1  1.61
## 5  1.97   0.222      1  1.45
## 6 -0.735  0.313      0 -0.568
```

Dealing with missing data: Replace missing data with the group mean

```
# replace
mcar <- mutate(mcar,
               mx1 = ifelse(is.na(x1), mean(x1, na.rm = TRUE), x1),
               mx2 = ifelse(is.na(x2), mean(x2, na.rm = TRUE), x2),
               mg  = ifelse(is.na( g), mean( g, na.rm = TRUE),  g))

# look at the relationship between x and y by g
(lm(y ~ mx1*mx2 + mg, data = mcar) %>%
   ci())[,1:3]
```

```
##               Estimate   CI lower  CI upper
## (Intercept) 0.09627262 -0.2298502 0.4223954
## mx1         1.04886558  0.7914195 1.3063117
## mx2         0.96273833  0.7191234 1.2063533
## mg          0.67035305  0.1804166 1.1602895
## mx1:mx2     1.38594940  1.1151227 1.6567761
```

Dealing with missing data: Impute

```r
require(missForest)
imp <- select(mcar, -mx1, -mx2, -mg) %>%
       as.data.frame() %>% # won't accept a tibble
       missForest()
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##   missForest iteration 5 in progress...done!
##   missForest iteration 6 in progress...done!
```

```r
(lm(y ~ x1*x2 + g, data = imp$ximp) %>%
  ci())[,1:3]
```

```
##                Estimate   CI lower  CI upper
## (Intercept) 0.01593081 -0.2585460 0.2904076
## x1          0.98954739  0.7748317 1.2042630
## x2          0.99484627  0.7939487 1.1957439
## g           0.73097516  0.3210541 1.1408962
## x1:x2       1.40140924  1.1741732 1.6286453
```

Missing at random

```r
set.seed(239847)
n <- 100
# generate a dataset with a lot of missing data
mar <- data_frame(x1 = rnorm(n),
                  x2 = rnorm(n),
                  g = rbinom(n, 1, .5),
                  y = x1 + x2 + x1*x2 + (g == 1) + rnorm(n)) %>%
       mutate(pmissing = ((y - min(y)) / sum(y - min(y)))^3 * 80000,
              x1 = ifelse(rbinom(n, 1, pmissing), NA, x1),
              x2 = ifelse(rbinom(n, 1, pmissing), NA, x2),
              g  = ifelse(rbinom(n, 1, pmissing), NA, g),
              y  = ifelse(rbinom(n, 1, pmissing), NA, y))
```

```
##                Estimate   CI lower  CI upper
## (Intercept) -0.1124975 -0.4384136 0.2134187
## x1           0.9969520  0.7111636 1.2827405
## x2           0.9867922  0.7367157 1.2368686
## g            0.7759519  0.2797262 1.2721777
## x1:x2        1.3225560  1.0557164 1.5893955
```

Missing at random - with imputation

```r
mar_imp <- as.data.frame(mar) %>% # won't accept a tibble
           missForest()
```

```r
(lm(y ~ x1*x2 + g, data = mar_imp$ximp) %>%
  ci())[,1:3]
```

```
##                Estimate   CI lower  CI upper
## (Intercept) -0.0144291 -0.2829760 0.2541178
## x1           0.9985559  0.7852141 1.2118977
## x2           0.9726862  0.7802398 1.1651325
## g            0.6474531  0.2484881 1.0464180
## x1:x2        1.3564751  1.1326662 1.5802840
```

Informative Missing

```r
set.seed(239847)
n <- 100
# generate a dataset with a lot of missing data
im <- data_frame(x1 = rnorm(n),
                 x2 = rnorm(n),
                 g = rbinom(n, 1, .5),
                 y = x1 + x2 + x1*x2 + (g == 1) + rnorm(n)) %>%
      mutate(pmissing = g / sum(g) * 20 * as.numeric(y > 0),
             x1 = ifelse(rbinom(n, 1, pmissing), NA, x1),
             x2 = ifelse(rbinom(n, 1, pmissing), NA, x2),
             g  = ifelse(rbinom(n, 1, pmissing), NA, g),
             y  = ifelse(rbinom(n, 1, pmissing), NA, y))
```

```
##                 Estimate   CI lower  CI upper
## (Intercept) -0.001682112 -0.2723151 0.2689509
## x1           1.088589510  0.8086144 1.3685647
## x2           0.975165408  0.7524343 1.1978965
## g            0.445831560 -0.1112362 1.0028993
## x1:x2        1.361745744  1.1264293 1.5970622
```

Informative missing - with imputation

```r
im_imp <- as.data.frame(im) %>% # won't accept a tibble
          missForest()
```

```r
(lm(y ~ x1*x2 + g, data = im_imp$ximp) %>%
  ci())[,1:3]
```

```
##                 Estimate   CI lower  CI upper
## (Intercept) -0.005479345 -0.2665632 0.2556045
## x1           1.045477133  0.8417941 1.2491602
## x2           1.076413317  0.8845495 1.2682771
## g            0.939877285  0.5561485 1.3236061
## x1:x2        1.299680810  1.0896144 1.5097472
```