IDS 702: Module 5.4

Multiple imputation in R

Dr. Olanrewaju Michael Akande

Illustration

- Simple example using data that come with the MICE package in R.
- Dataset from NHANES includes 25 cases measured on 4 variables.
- Only 13 cases with complete data.
- We will use multiple imputation to make completed datasets and do analyses.
- The four variables are
    1. age (age group: 20-39, 40-59, 60+)
    2. bmi (body mass index, in kg/m^2)
    3. hyp (hypertension status: no, yes)
    4. chl (total cholesterol, in mg/dL)

Illustration

```r
library(mice)
data(nhanes2)
dim(nhanes2)
```

```
## [1] 25  4
```

```r
summary(nhanes2)
```

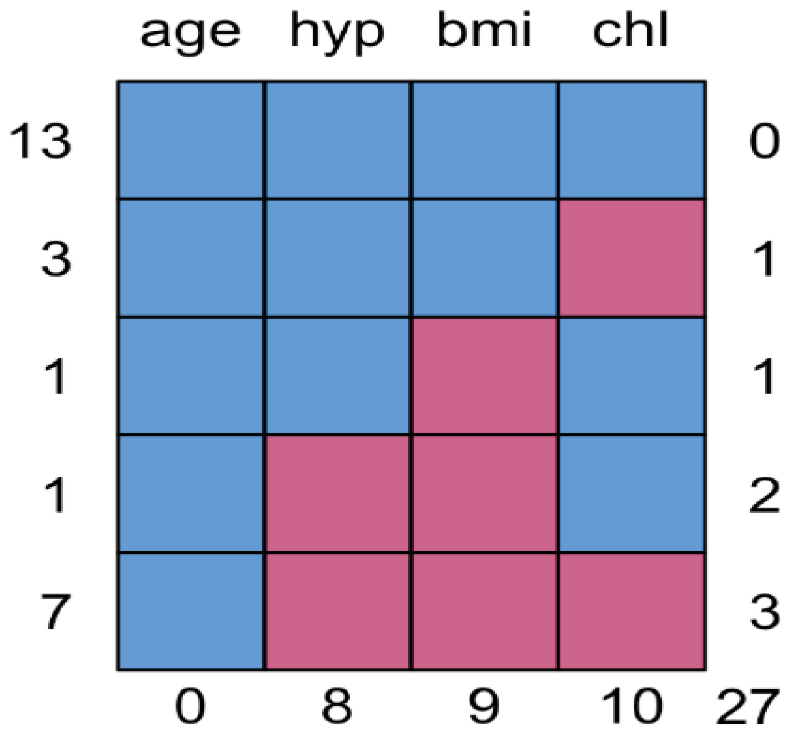
```
##     age          bmi          hyp          chl
##  20-39:12   Min.   :20.40   no  :13   Min.   :113.0
##  40-59: 7   1st Qu.:22.65   yes : 4   1st Qu.:185.0
##  60-99: 6   Median :26.75   NA's: 8   Median :187.0
##             Mean   :26.56             Mean   :191.4
##             3rd Qu.:28.93             3rd Qu.:212.0
##             Max.   :35.30             Max.   :284.0
##             NA's   :9                 NA's   :10
```

```r
str(nhanes2)
```

```
## 'data.frame':    25 obs. of  4 variables:
##  $ age: Factor w/ 3 levels "20-39","40-59",..: 1 2 1 3 1 3 1 1 2 2 ...
##  $ bmi: num  NA 22.7 NA NA 20.4 NA 22.5 30.1 22 NA ...
##  $ hyp: Factor w/ 2 levels "no","yes": NA 1 1 NA 1 NA 1 1 1 NA ...
##  $ chl: num  NA 187 187 NA 113 184 118 187 238 NA ...
```

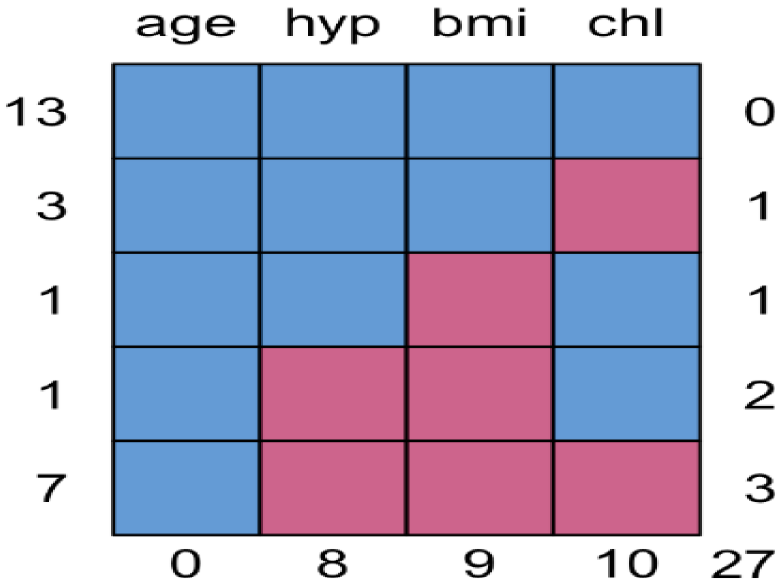
Patterns of missing data

```
md.pattern(nhanes2)
```



5 patterns observed from $2^3 = 8$ possible patterns

Patterns of missing data



- At the bottom: total number of missing values by variables.
- On the right: number of variables missing in each pattern.
- On the left: number of cases for each pattern.

Visualizing patterns of missing data

```r
library(VIM); library(lattice)
aggr(nhanes2,col=c("lightblue3","darkred"),numbers=TRUE,sortVars=TRUE,
     labels=names(nhanes2),cex.axis=.7,gap=3,
     ylab=c("Proportion missing","Missingness pattern"))
```

```
## 
##  Variables sorted by number of missings: 
##  Variable Count
##       chl  0.40
##       bmi  0.36
##       hyp  0.32
##       age  0.00
```

Visualizing patterns of missing data

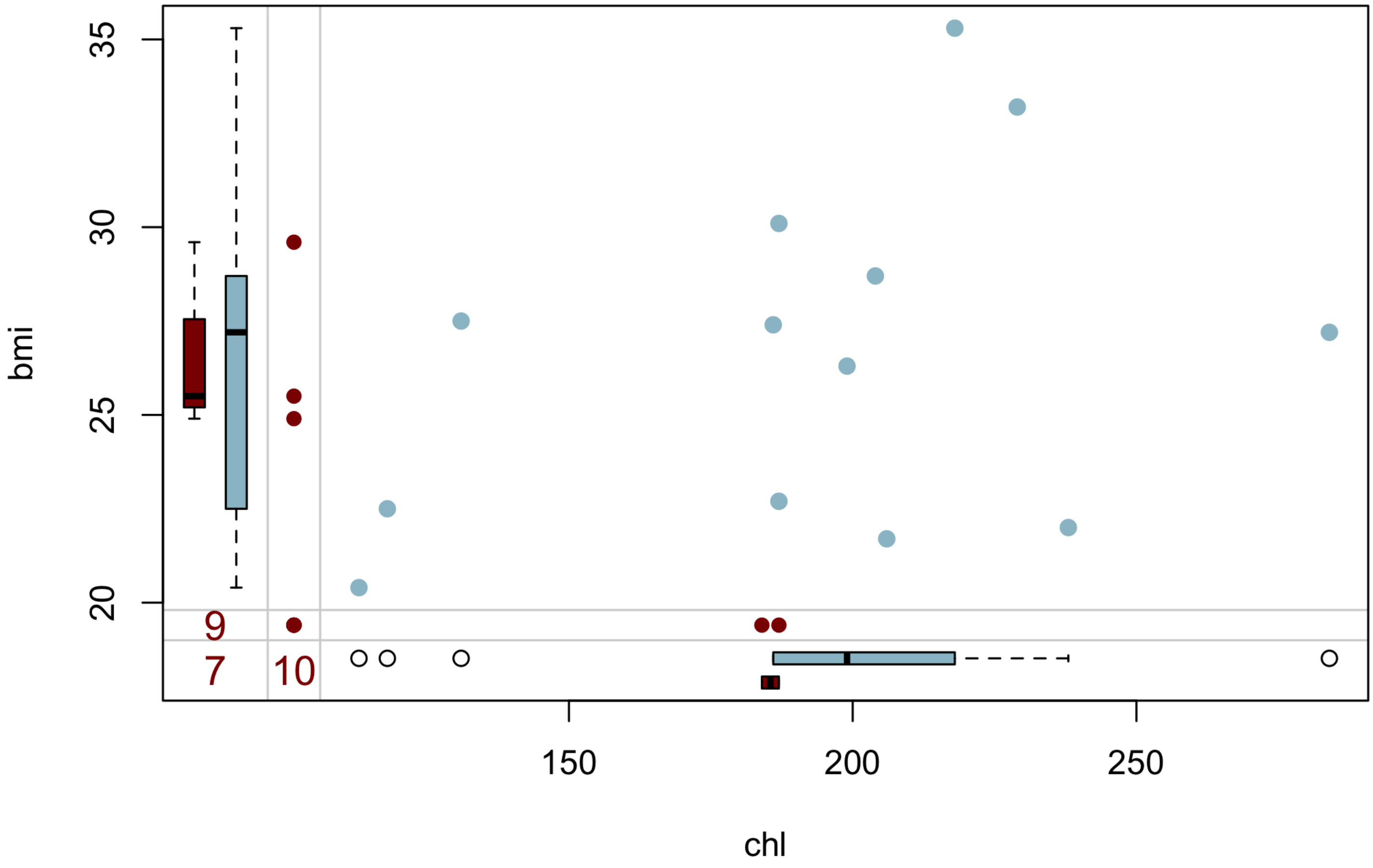
The marginplot function can be used to understand how missingness affects the distribution of values on other variables.

- Blue box plots summarize the distribution of observed data given the other variable is observed.
- Red box plots summarize the distribution of observed data given the other variable is missing.
- If data are MCAR, you expect the boxplots to be the same (hard to evaluate in this small sample)

Let's look at the margin plot for the two continuous variables `bmi` and `chl`.

Visualizing patterns of missing data

```
marginplot(nhanes2[,c("chl","bmi")],col=c("lightblue3","darkred"),cex.numbers=1.2,pch=19)
```



Visualizing patterns of missing data

- Interpretation of the numbers in red.
  - 9 = number of observations with missingness in `bmi`
  - 10 = number of observations with missingness in `chl`
  - 7 = number of observations with missingness in both `bmi` and `chl`.
- The scatterplot of blue points display the relationship between `bmi` and `chl` when they are both observed (13 cases).
- The red points indicate the amount of data used to generate the red boxplots.

Mice in R

We will use the mice function to generate 10 imputed datasets. By default, mice uses

- pmm: Predictive mean matching for numeric data
- logreg: Logistic regression for factor data with 2 levels
- polyreg: Multinomial logistic regression for factor data with > 2 levels
- polr: Proportional odds model for factor data with > 2 ordered levels

Mice in R

Other commonly used methods are

- norm: Bayesian linear regression
- sample: Random sample from observed values
- cart: Classification and regression trees
- rf: Random forest

Personally, I prefer to use norm instead of pmm for imputing numeric/continuous variables.

For the illustration,

```r
nhanes2_imp <- mice(nhanes2,m=10,
                    defaultMethod=c("norm","logreg","polyreg","polr"),
                    print=F)
```

Mice in R

```r
methods(mice)
```

```
## Warning in .S3methods(generic.function, class, envir): function 'mice' appears
## not to be S3 generic; found functions that look like S3 methods

##  [1] mice.impute.2l.bin       mice.impute.2l.lmer      mice.impute.2l.norm
##  [4] mice.impute.2l.pan       mice.impute.2lonly.mean  mice.impute.2lonly.norm
##  [7] mice.impute.2lonly.pmm   mice.impute.cart         mice.impute.jomoImpute
## [10] mice.impute.lda          mice.impute.logreg       mice.impute.logreg.boot
## [13] mice.impute.mean         mice.impute.midastouch   mice.impute.mnar.logreg
## [16] mice.impute.mnar.norm    mice.impute.norm         mice.impute.norm.boot
## [19] mice.impute.norm.nob     mice.impute.norm.predict mice.impute.panImpute
## [22] mice.impute.passive      mice.impute.pmm          mice.impute.polr
## [25] mice.impute.polyreg      mice.impute.quadratic    mice.impute.rf
## [28] mice.impute.ri           mice.impute.sample       mice.mids
## [31] mice.theme
## see '?methods' for accessing help and source code
```

Predictive mean matching (pmm)

- Suppose y is subject to missing values while x is completely observed. The basic idea for pmm is:
  - Using complete cases, regress y on x, obtaining $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1)$;
  - Draw a new $\beta^\star$ from the "posterior distribution" of $\hat{\beta}$ (e.g, multivariate normal);
  - Using $\beta^\star$, generate predicted values of y for all cases;
  - For each case with a missing y, identify set of donors with no missing values, who have predicted y values close to that of the case with missing data;
  - From among these cases, randomly select one and assign its observed value of y as the imputed value;
  - Repeat for all observations and imputation data sets.
- Pmm matches the distribution of the original observed variable, as imputed values are taken from the real data.

Mice in R

Back to the `nhanes2_imp` object, first look at the original data

```
nhanes2
```

```
##      age  bmi  hyp chl
## 1  20-39   NA <NA>  NA
## 2  40-59 22.7   no 187
## 3  20-39   NA   no 187
## 4  60-99   NA <NA>  NA
## 5  20-39 20.4   no 113
## 6  60-99   NA <NA> 184
## 7  20-39 22.5   no 118
## 8  20-39 30.1   no 187
## 9  40-59 22.0   no 238
## 10 40-59   NA <NA>  NA
## 11 20-39   NA <NA>  NA
## 12 40-59   NA <NA>  NA
## 13 60-99 21.7   no 206
## 14 40-59 28.7  yes 204
## 15 20-39 29.6   no  NA
## 16 20-39   NA <NA>  NA
## 17 60-99 27.2  yes 284
## 18 40-59 26.3  yes 199
## 19 20-39 35.3   no 218
## 20 60-99 25.5  yes  NA
## 21 20-39   NA <NA>  NA
## 22 20-39 33.2   no 229
## 23 20-39 27.5   no 131
## 24 60-99 24.9   no  NA
## 25 40-59 27.4   no 186
```

Mice in R

Look at the first imputed-dataset

```
d1 <-  complete(nhanes2_imp, 1); d1
```

```
##      age      bmi hyp      chl
## 1  20-39 30.34098  no 224.2285
## 2  40-59 22.70000  no 187.0000
## 3  20-39 27.29756  no 187.0000
## 4  60-99 35.61975 yes 297.0469
## 5  20-39 20.40000  no 113.0000
## 6  60-99 18.16587 yes 184.0000
## 7  20-39 22.50000  no 118.0000
## 8  20-39 30.10000  no 187.0000
## 9  40-59 22.00000  no 238.0000
## 10 40-59 27.84743  no 210.5014
## 11 20-39 27.24996  no 146.9218
## 12 40-59 28.43579  no 226.0825
## 13 60-99 21.70000  no 206.0000
## 14 40-59 28.70000 yes 204.0000
## 15 20-39 29.60000  no 208.7224
## 16 20-39 36.73795  no 230.7358
## 17 60-99 27.20000 yes 284.0000
## 18 40-59 26.30000 yes 199.0000
## 19 20-39 35.30000  no 218.0000
## 20 60-99 25.50000 yes 257.7126
## 21 20-39 22.42268  no 127.4948
## 22 20-39 33.20000  no 229.0000
## 23 20-39 27.50000  no 131.0000
## 24 60-99 24.90000  no 283.3828
## 25 40-59 27.40000  no 186.0000
```

IDS 702

Mice in R

Look at the last imputed-dataset

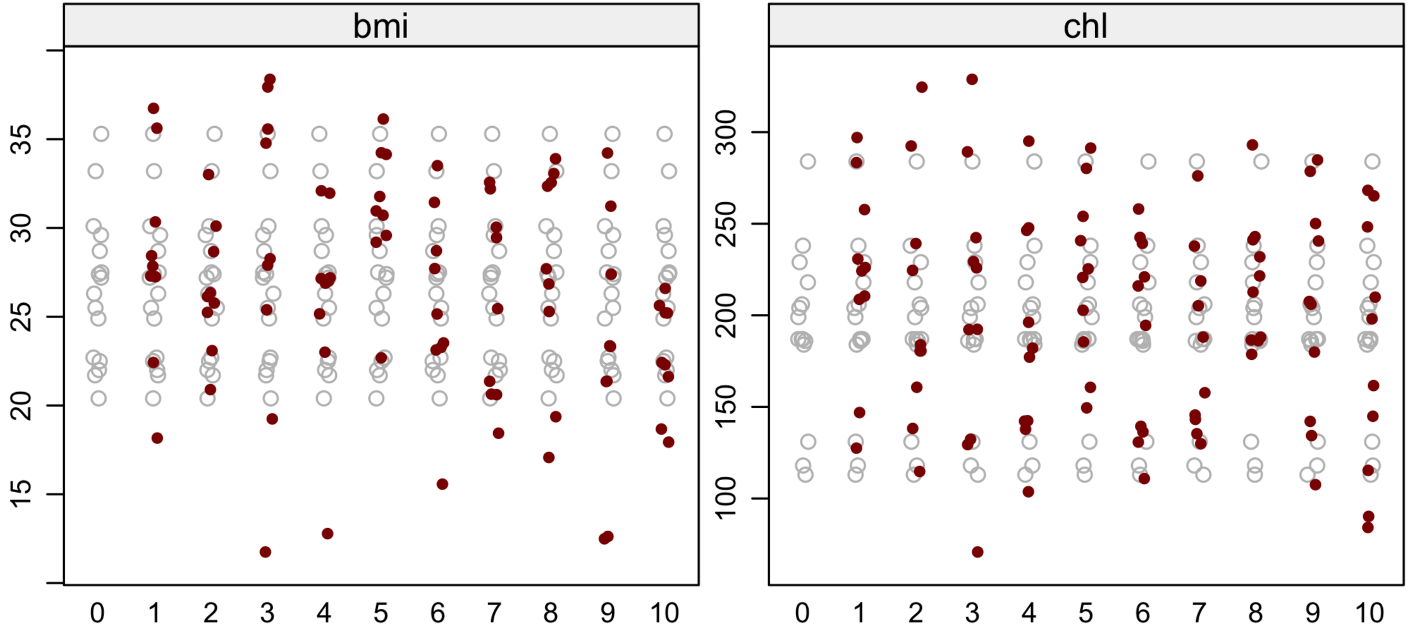
```
d10 <-  complete(nhanes2_imp, 10); d10
```

```
##      age      bmi hyp       chl
## 1  20-39 26.59731  no 115.34762
## 2  40-59 22.70000  no 187.00000
## 3  20-39 22.42548  no 187.00000
## 4  60-99 25.20745 yes 268.22285
## 5  20-39 20.40000  no 113.00000
## 6  60-99 22.29470  no 184.00000
## 7  20-39 22.50000  no 118.00000
## 8  20-39 30.10000  no 187.00000
## 9  40-59 22.00000  no 238.00000
## 10 40-59 25.63055 yes 198.06595
## 11 20-39 17.93695  no  90.22238
## 12 40-59 21.62430 yes 209.92840
## 13 60-99 21.70000  no 206.00000
## 14 40-59 28.70000 yes 204.00000
## 15 20-39 29.60000  no 161.64022
## 16 20-39 18.67313  no  84.17346
## 17 60-99 27.20000 yes 284.00000
## 18 40-59 26.30000 yes 199.00000
## 19 20-39 35.30000  no 218.00000
## 20 60-99 25.50000 yes 265.20195
## 21 20-39 25.22149  no 144.88429
## 22 20-39 33.20000  no 229.00000
## 23 20-39 27.50000  no 131.00000
## 24 60-99 24.90000  no 248.27329
## 25 40-59 27.40000  no 186.00000
```

Illustration

Let's plot imputed and observed values for continuous variables.

```
stripplot(nhanes2_imp, col=c("grey","darkred"),pch=c(1,20))
```

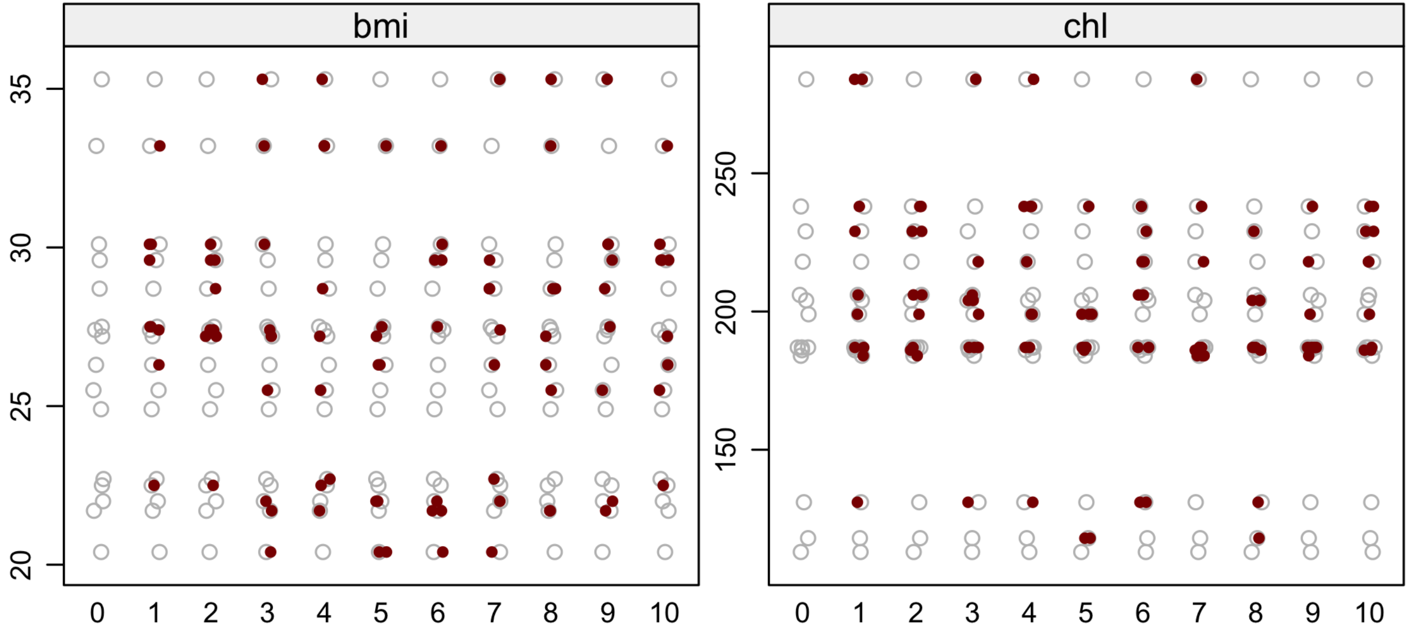


Grey dots are observed values and red dots are imputed values.

Illustration

Let's see how this would change when we use `pmm` instead of `norm`.

```
nhanes2_imp2 <- mice(nhanes2,m=10,defaultMethod=c("pmm","logreg","polyreg","polr"),print=F
stripplot(nhanes2_imp2, col=c("grey","darkred"),pch=c(1,20))
```

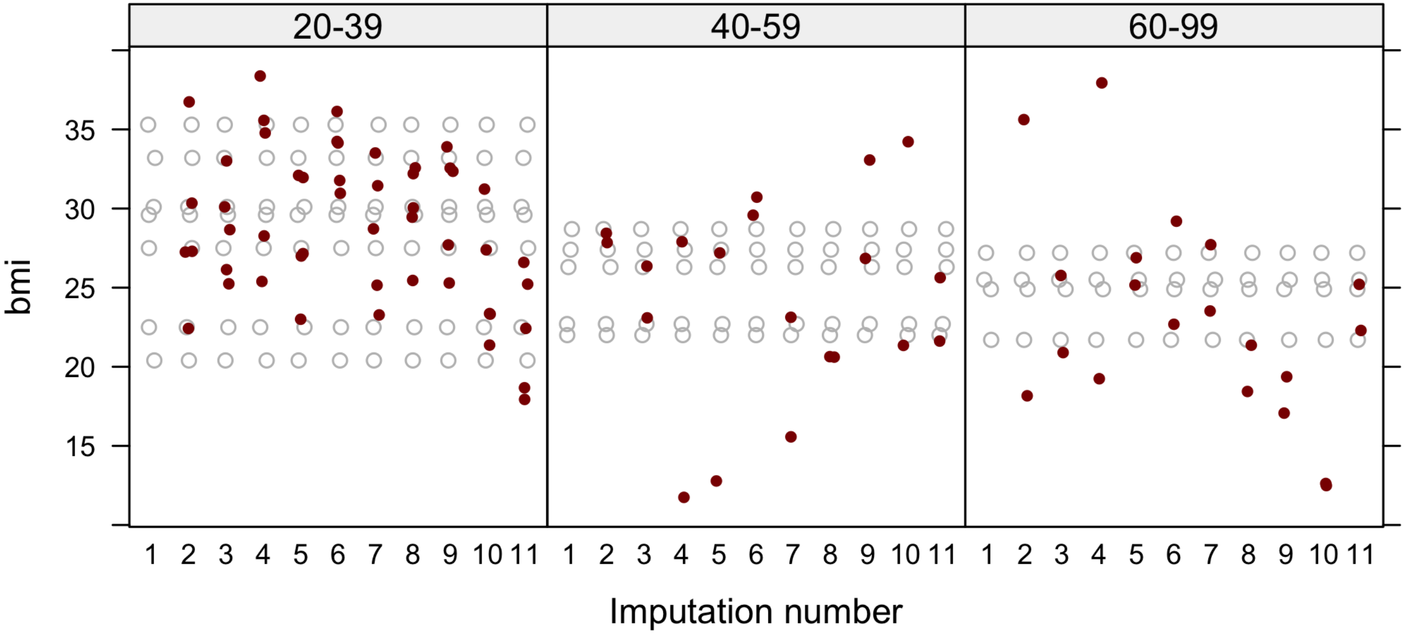


Easy to see that the distribution of the original observed data is preserved.

ILLUSTRATION

Also can do plots by values of categorical variable, say `bmi` by `age`. Let's look at the imputations using `norm`

```
stripplot(nhanes2_imp, bmi~.imp|age, col=c("grey","darkred"),pch=c(1,20))
```

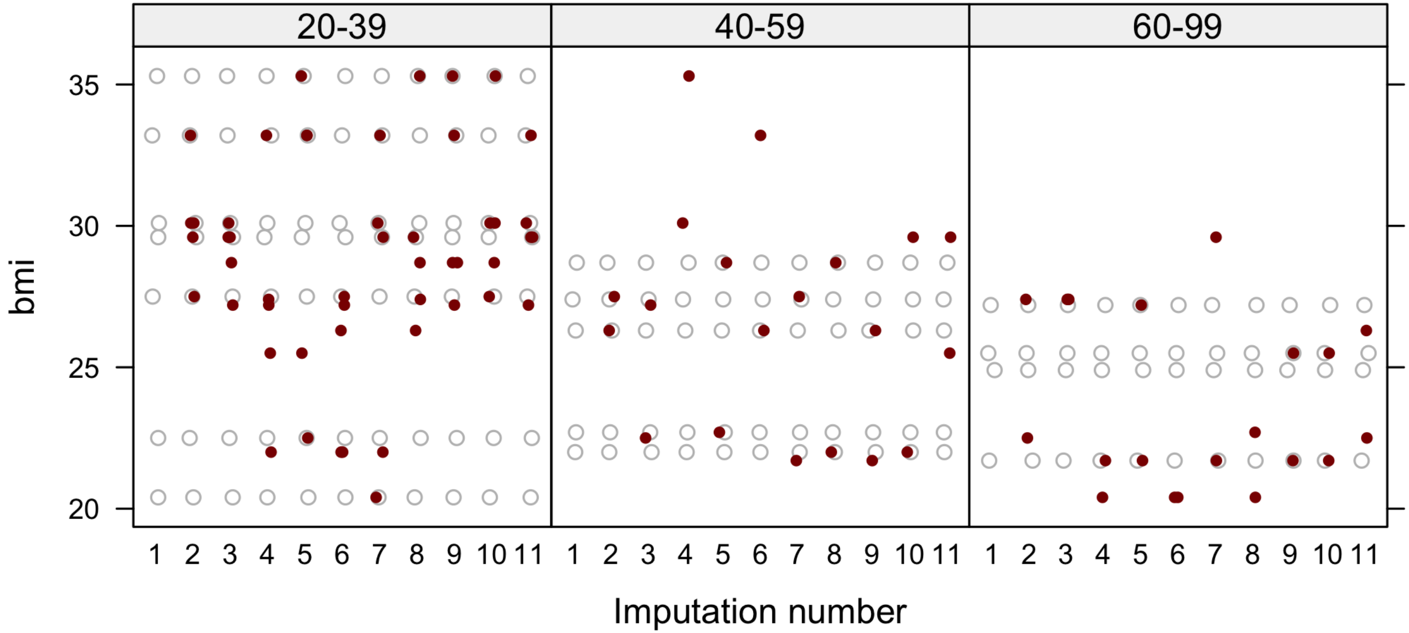


IDS 702

Illustration

Using `pmm` instead of `norm`, we have:

```
stripplot(nhanes2_imp2, bmi~.imp|age, col=c("grey","darkred"),pch=c(1,20))
```

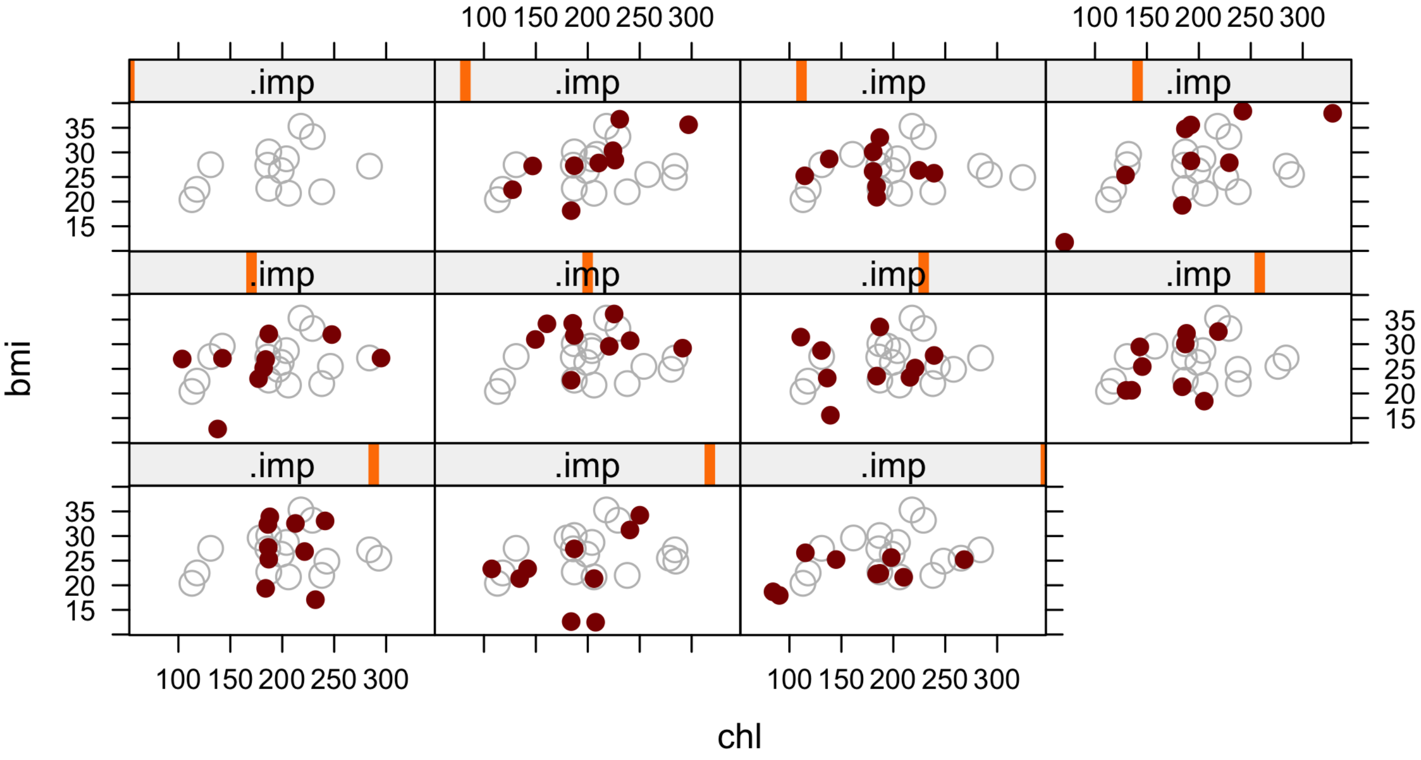


Going forward, let's focus only on imputations using `norm`.

ILLUSTRATION

Scatterplot of `chl` and `bmi` for each imputed dataset. Here we can see why we should not use single imputations.

```
xyplot(nhanes2_imp, bmi ~ chl | .imp,pch=c(1,20),cex = 1.4,col=c("grey","darkred"))
```



IDS 702

Illustration

To detect interesting differences in distribution between observed and imputed data, use the densityplot function.

```
densityplot(nhanes2_imp)
```

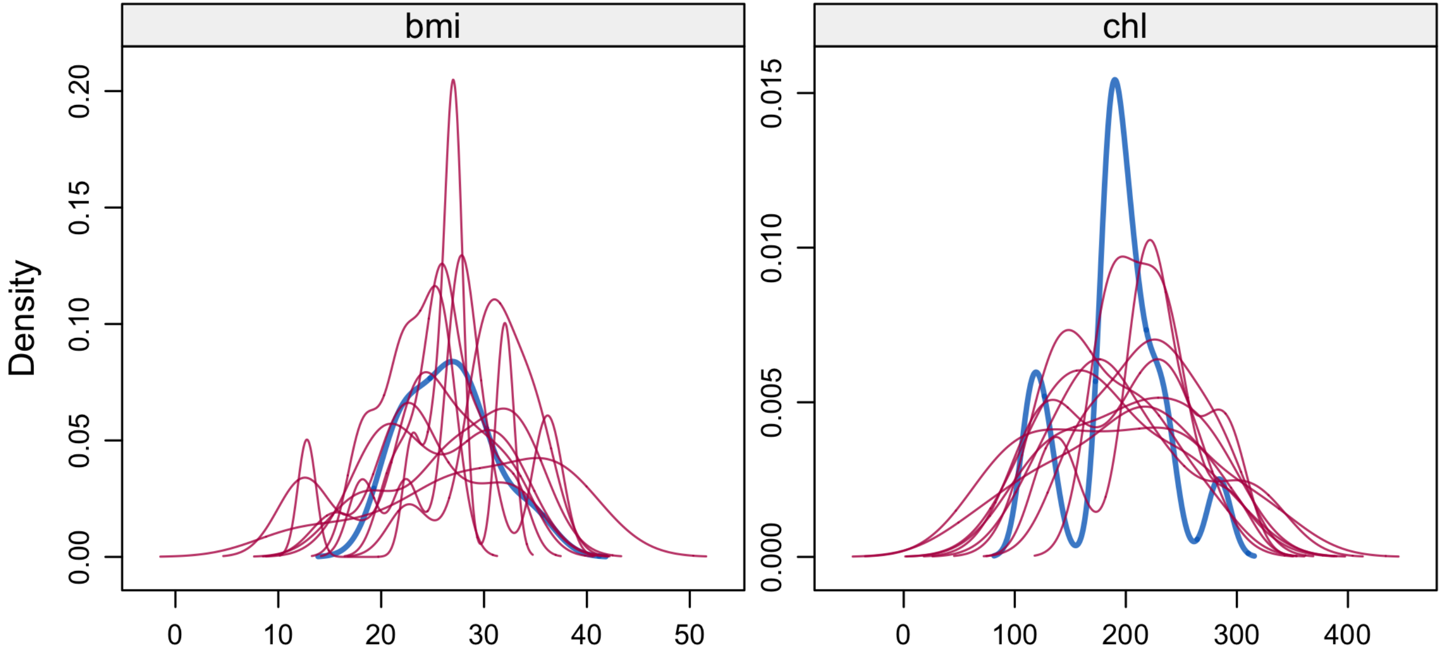


Illustration: using a single dataset

For model specification, i.e., transformations, either look at the complete cases or use one of the completed datasets. For example, to use the first dataset in a regression of `bmi` on `age`, `hyp` and `chl`, use

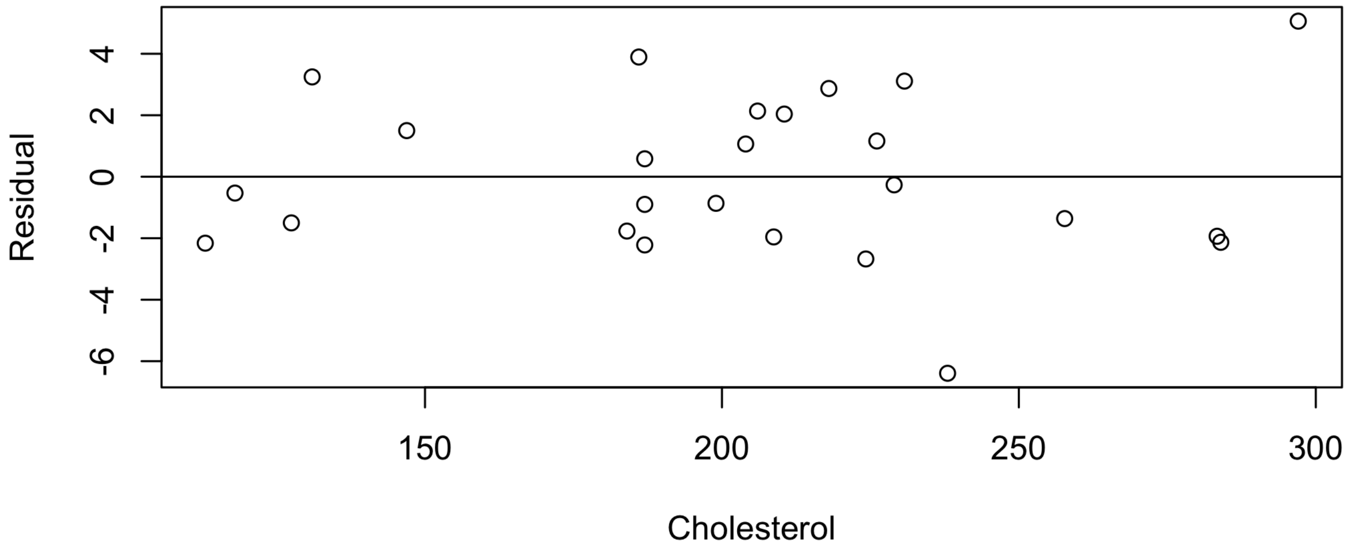
```
bmiregd1 <- lm(bmi~age+hyp+chl, data = d1)
summary(bmiregd1)
```

```
##
## Call:
## lm(formula = bmi ~ age + hyp + chl, data = d1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3936 -1.9368 -0.5314  2.0384  5.0614
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.94077    2.74697   4.347 0.000313 ***
## age40-59     -5.91646    1.50677  -3.927 0.000835 ***
## age60-99    -11.73873    2.13243  -5.505 2.18e-05 ***
## hypyes        2.43724    1.71224   1.423 0.170027
## chl           0.09399    0.01483   6.338 3.48e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.845 on 20 degrees of freedom
## Multiple R-squared:  0.7062,    Adjusted R-squared:  0.6474
## F-statistic: 12.02 on 4 and 20 DF,  p-value: 3.866e-05
```

ILLUSTRATION: USING A SINGLE DATASET

- To check residuals, you can examine the fit of the model in one or more completed datasets
- Any transformations will have to apply to all the datasets, so don't be too dataset-specific in your checks.

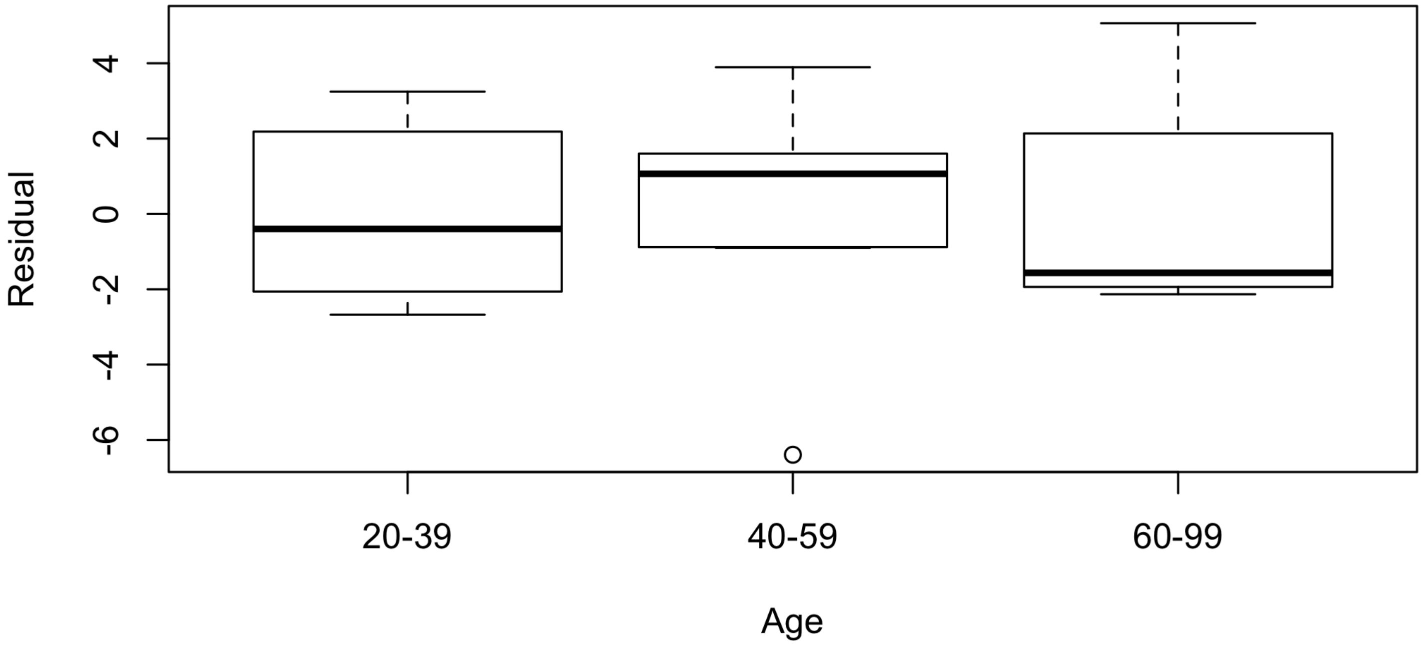
```
plot(bmiregd1$residual,x=d1$chl,xlab="Cholesterol",ylab="Residual"); abline(0,0)
```



Looks good!

Illustration: using a single dataset

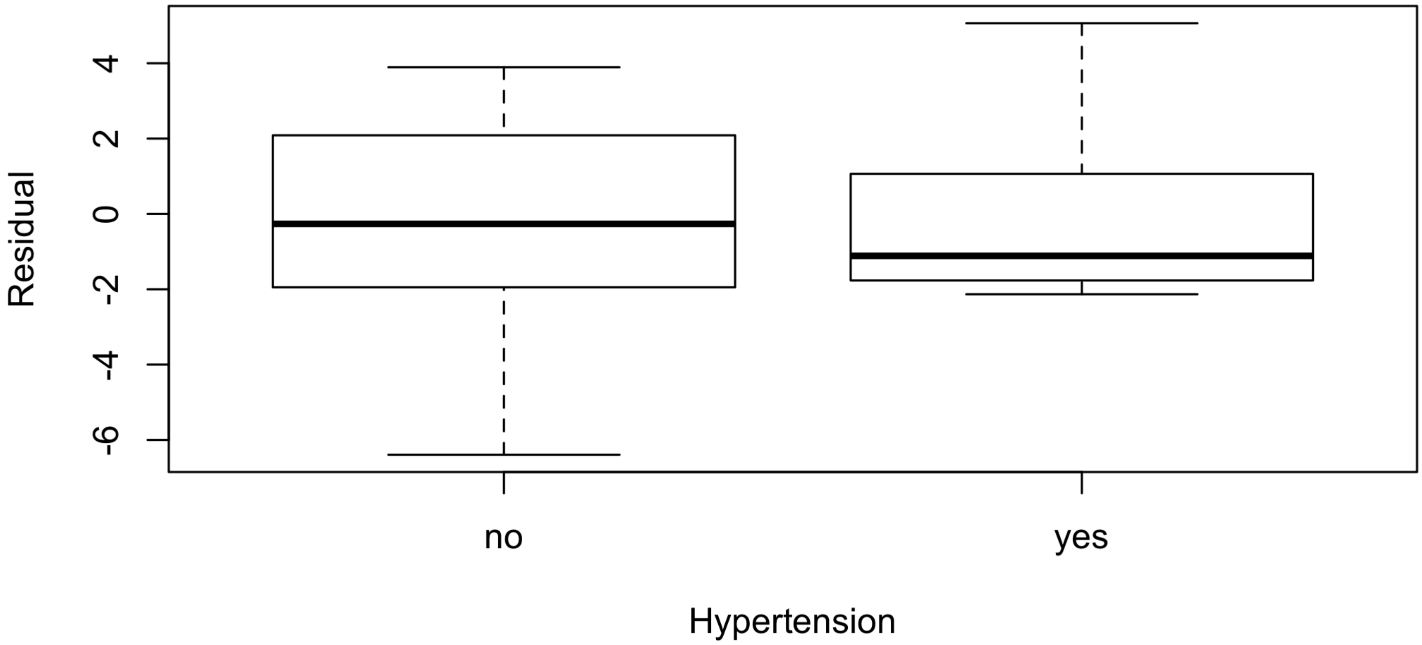
```
boxplot(bmiregd1$residual ~ d1$age, xlab = "Age", ylab = "Residual")
```



Pretty reasonable especially given the size of the dataset.

Illustration: using a single dataset

```r
boxplot(bmiregd1$residual ~ d1$hyp, xlab = "Hypertension", ylab = "Residual")
```



- Good idea to repeat for more than one completed dataset.
- If you decide transformations are needed, you might reconsider the imputation models too and fit them with transformed values.

Illustration: using all M datasets

```
bmireg_imp <- with(data=nhanes2_imp, lm(bmi~age+hyp+chl))
#results for second dataset
bmireg_imp[[4]][[2]]
```

```
##
## Call:
## lm(formula = bmi ~ age + hyp + chl)
##
## Coefficients:
## (Intercept)     age40-59     age60-99       hypyes          chl
##    18.44372     -6.53144    -10.51157      1.41484      0.06152
```

```
#results for fifth dataset
bmireg_imp[[4]][[5]]
```

```
##
## Call:
## lm(formula = bmi ~ age + hyp + chl)
##
## Coefficients:
## (Intercept)     age40-59     age60-99       hypyes          chl
##    18.17017     -7.50997    -11.66474      2.31496      0.07016
```

IDS 702

Illustration: using all M datasets

Now to get the multiple imputation inferences based on the Rubin (1987) combining rules

```
bmireg <- pool(bmireg_imp)
summary(bmireg)
```

```
##          term     estimate  std.error  statistic        df     p.value
## 1 (Intercept)  16.03269801 4.37392771  3.6655151  6.317986 0.009594146
## 2    age40-59  -6.57207247 1.97111257 -3.3341944 11.653096 0.006179856
## 3    age60-99 -10.83536538 2.90742365 -3.7267928  7.366405 0.006740526
## 4      hypyes   2.04370148 2.35795119  0.8667276  8.976479 0.408661074
## 5         chl   0.07394739 0.02428787  3.0446227  6.597552 0.020136806
```

Illustration: using all M datasets

- You can still do a nested F test (well, technically a test that is asymptotically equivalent to a nested F test) for the multiply-imputed dataset using the pool.compare function.

- For example, suppose we want to see if age is a useful predictor, then

```
bmireg_imp <- with(data=nhanes2_imp, lm(bmi~hyp+chl+age))
bmireg_impnoage <- with(data=nhanes2_imp, lm(bmi~hyp+chl))
#type "pool.compare(bmireg_imp, bmireg_impnoage)" to see full results
pool.compare(bmireg_imp, bmireg_impnoage)[c(9:12,18)]
```

```
## $qbar1
##  (Intercept)       hypyes          chl     age40-59     age60-99
##  16.03269801   2.04370148   0.07394739  -6.57207247 -10.83536538
##
## $qbar0
## (Intercept)      hypyes         chl
## 20.78380117 -0.88512898  0.03078513
##
## $ubar1
## [1] 8.4661183053 3.2824280668 0.0002711616 2.7734500763 4.2665578296
##
## $ubar0
## [1] 1.738379e+01 6.590985e+00 4.824345e-04
##
## $pvalue
##              [,1]
## [1,] 1.654266e-05
```

Illustration: using all M datasets

You also can fit logistic regressions. For example to predict hypertension from all the other variables, do

```
hyplogreg_imp <- with(data=nhanes2_imp, glm(hyp~bmi+chl+age, family = binomial))
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

This turns out to be problematic here because we have some logistic regressions with perfect predictions.

Illustration: using all M datasets

```r
hyplogreg <- pool(hyplogreg_imp)
summary(hyplogreg)
```

```
##          term     estimate   std.error     statistic       df   p.value
## 1 (Intercept) -5042.073708  788097.633 -6.397778e-03 18.25726 0.9949647
## 2         bmi    90.415352   13854.295  6.526161e-03 18.25698 0.9948637
## 3         chl     8.402631    1359.481  6.180766e-03 18.25763 0.9951355
## 4    age40-59   996.882034  169469.486  5.882369e-03 18.25787 0.9953704
## 5    age60-99   503.458547 8664479.986  5.810603e-05 18.25904 0.9999543
```

We do not have enough data to do a meaningful logistic regression here, unless we drop age as a predictor, but the command structure is fine!

Modifying predictors in the imputation models

Going back to the imputed datasets, which variables does `mice()` use as predictors for imputation of each incomplete variable?

```
nhanes2_imp$predictorMatrix
```

```
##     age bmi hyp chl
## age   0   1   1   1
## bmi   1   0   1   1
## hyp   1   1   0   1
## chl   1   1   1   0
```

Modifying predictors in the imputation models

We can choose to exclude variables from any of the imputation models. For example, suppose we think that `hyp` should not predict `bmi`. Then,

```
pred <- nhanes2_imp$predictorMatrix
pred["bmi","hyp"] <- 0
pred
```

```
##     age bmi hyp chl
## age   0   1   1   1
## bmi   1   0   0   1
## hyp   1   1   0   1
## chl   1   1   1   0
```

```
mice(nhanes2,m=10,defaultMethod=c("norm","logreg","polyreg","polr"),predictorMatrix=pred)
```

What's next?

Move on to the readings for the next module!