Missing Data
EM Algorithm and Multiple Imputation

Aaron Molstad, Dootika Vats, Li Zhong

University of Minnesota
School of Statistics

December 4, 2013
Overview

1. EM Algorithm

2. Multiple Imputation
Incomplete Data

- Consider two sample spaces $\mathcal{Y}$ and $\mathcal{X}$
Incomplete Data

- Consider two sample spaces $\mathcal{Y}$ and $\mathcal{X}$
- The observed data $y$ are a realization from $\mathcal{Y}$
Incomplete Data

- Consider two sample spaces $Y$ and $X$
- The observed data $y$ are a realization from $Y$
- The corresponding $x$ in $X$ is not observable
Incomplete Data

- Consider two sample spaces $\mathcal{Y}$ and $\mathcal{X}$
- The observed data $y$ are a realization from $\mathcal{Y}$
- The corresponding $x$ in $\mathcal{X}$ is not observable
- A map $F: \mathcal{Y} \rightarrow \mathcal{X}$
- The preimage $F^{-1}(y)$ is called the germ at $y$
Consider two sample spaces $\mathcal{Y}$ and $\mathcal{X}$

The observed data $y$ are a realization from $\mathcal{Y}$

The corresponding $x$ in $\mathcal{X}$ is not observable

A map $F: \mathcal{Y} \rightarrow \mathcal{X}$

The preimage $F^{-1}(y)$ is called the germ at $y$

$x$ includes data and parameters
EM Algorithm

- \( f(x|\phi) \) is a family of sampling densities, and

\[
g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx
\]
**EM Algorithm**

- $f(x|\phi)$ is a family of sampling densities, and

\[
g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx
\]

- The EM algorithm aims to find a $\phi$ that maximizes $g(y|\phi)$ given an observed $y$, while making essential use of $f(x|\phi)$
EM Algorithm

- $f(x|\phi)$ is a family of sampling densities, and

\[
g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx
\]

- The EM algorithm aims to find a $\phi$ that maximizes $g(y|\phi)$ given an observed $y$, while making essential use of $f(x|\phi)$

- Each iteration includes two steps:
EM Algorithm

- \( f(x|\phi) \) is a family of sampling densities, and

\[
g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx
\]

- The EM algorithm aims to find a \( \phi \) that maximizes \( g(y|\phi) \) given an observed \( y \), while making essential use of \( f(x|\phi) \)

- Each iteration includes two steps:

- The expectation step (E-step) uses current estimate of the parameter to find (expectation of) complete data
EM Algorithm

- $f(x|\phi)$ is a family of sampling densities, and

$$g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx$$

- The EM algorithm aims to find a $\phi$ that maximizes $g(y|\phi)$ given an observed $y$, while making essential use of $f(x|\phi)$

- Each iteration includes two steps:

  - The expectation step (E-step) uses current estimate of the parameter to find (expectation of) complete data
  - The maximization step (M-step) uses the updated data from the E-step to find a maximum likelihood estimate of the parameter

Stop the algorithm when change of estimated parameter reaches a preset threshold.
EM Algorithm

- \( f(x|\phi) \) is a family of sampling densities, and

\[
g(y|\phi) = \int_{F^{-1}(y)} f(x|\phi) \, dx
\]

- The EM algorithm aims to find a \( \phi \) that maximizes \( g(y|\phi) \) given an observed \( y \), while making essential use of \( f(x|\phi) \)

- Each iteration includes two steps:
  - The expectation step (E-step) uses current estimate of the parameter to find (expectation of) complete data
  - The maximization step (M-step) uses the updated data from the E-step to find a maximum likelihood estimate of the parameter

- Stop the algorithm when change of estimated parameter reaches a preset threshold.
A Multinomial Example

Consider data from Rao(1965) with 197 animals multinomially distributed in four categories:

\[ y = (y_1, y_2, y_3, y_4) = (125, 18, 20, 34) \]

A genetic model specifies cell probabilities:

\[ \left( \frac{1}{2} + \frac{1}{4}\pi, \frac{1}{4}(1 - \pi), \frac{1}{4}(1 - \pi), \frac{1}{4}\pi \right) \]
A Multinomial Example

Consider data from Rao(1965) with 197 animals multinomially distributed in four categories:

\[ y = (y_1, y_2, y_3, y_4) = (125, 18, 20, 34) \]

A genetic model specifies cell probabilities:

\[ \left( \frac{1}{2} + \frac{1}{4} \pi, \frac{1}{4} (1 - \pi), \frac{1}{4} (1 - \pi), \frac{1}{4} \pi \right) \]

\[ g(y|\pi) = \frac{(y_1 + y_2 + y_3 + y_4)!}{y_1! y_2! y_3! y_4!} \left( \frac{1}{2} + \frac{1}{4} \pi \right)^{y_1} \left( \frac{1}{4} (1 - \pi) \right)^{y_2} \left( \frac{1}{4} (1 - \pi) \right)^{y_3} \left( \frac{1}{4} \pi \right)^{y_4} \]
A Multinomial Example: continued

Complete data: a multinomial population

\[ \mathbf{x} = (x_1, x_2, x_3, x_4, x_5) \]
Complete data: a multinomial population

\[ x = (x_1, x_2, x_3, x_4, x_5) \]

Cell probabilities:

\[ \left( \frac{1}{2}, \frac{1}{4}\pi, \frac{1}{4}(1 - \pi), \frac{1}{4}(1 - \pi), \frac{1}{4}\pi \right) \]
A Multinomial Example: continued

Complete data: a multinomial population

\[ \mathbf{x} = (x_1, x_2, x_3, x_4, x_5) \]

Cell probabilities:

\[ \left( \frac{1}{2}, \frac{1}{4} \pi, \frac{1}{4}(1 - \pi), \frac{1}{4}(1 - \pi), \frac{1}{4} \pi \right) \]

\[
f(\mathbf{x}|\pi) = \frac{(x_1 + x_2 + x_3 + x_4 + x_5)!}{x_1!x_2!x_3!x_4!x_5!} \left( \frac{1}{2} \right)^{x_1} \left( \frac{1}{4} \pi \right)^{x_2} \left( \frac{1}{4} - \frac{1}{4} \pi \right)^{x_3} \left( \frac{1}{4} - \frac{1}{4} \pi \right)^{x_4} \left( \frac{1}{4} \pi \right)^{x_5} \]

Next we will show how EM algorithm works in this example.
Complete data: a multinomial population

\[ \mathbf{x} = (x_1, x_2, x_3, x_4, x_5) \]

Cell probabilities:

\[ \left( \frac{1}{2}, \frac{1}{4}\pi, \frac{1}{4}(1 - \pi), \frac{1}{4}(1 - \pi), \frac{1}{4}\pi \right) \]

\[
f(\mathbf{x}|\pi) = \frac{(x_1 + x_2 + x_3 + x_4 + x_5)!}{x_1!x_2!x_3!x_4!x_5!} \left(\frac{1}{2}\right)^{x_1} \left(\frac{1}{4}\pi\right)^{x_2} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{x_3} \left(\frac{1}{4} - \frac{1}{4}\pi\right)^{x_4} \left(\frac{1}{4}\pi\right)^{x_5}\]

Next we will show how EM algorithm works in this example.
Let $\pi^{(p)}$ be the value of $\pi$ after $p$ iterations.

$(x_3, x_4, x_5)$ are fixed in this example.

$x_1 + x_2 = y_1 = 125$ and $\pi = \pi^{(p)}$ gives
Let $\pi^{(p)}$ be the value of $\pi$ after $p$ iterations.

$(x_3, x_4, x_5)$ are fixed in this example.

$x_1 + x_2 = y_1 = 125$ and $\pi = \pi^{(p)}$ gives

$$x_1^{(p)} = 125 \cdot \frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{4} \pi^{(p)}},$$

$$x_2^{(p)} = 125 \cdot \frac{\frac{1}{4} \pi^{(p)}}{\frac{1}{2} + \frac{1}{4} \pi^{(p)}}$$
Let $\pi^{(p)}$ be the value of $\pi$ after $p$ iterations.

$(x_3, x_4, x_5)$ are fixed in this example.

$x_1 + x_2 = y_1 = 125$ and $\pi = \pi^{(p)}$ gives

$$x_1^{(p)} = 125 \cdot \frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{4}\pi(p)}, \quad x_2^{(p)} = 125 \cdot \frac{\frac{1}{4}\pi(p)}{\frac{1}{2} + \frac{1}{4}\pi(p)}$$

The next step will use the complete data estimated in this step.
We use \((x_1^{(p)}, x_2^{(p)}, 18, 20, 34)\) as if these estimated data were the observed data, and find the maximum likelihood estimate of \(\pi\), denoted \(\pi^{(p+1)}\).
We use \((x_1^{(p)}, x_2^{(p)}, 18, 20, 34)\) as if these estimated data were the observed data, and find the maximum likelihood estimate of \(\pi\), denoted \(\pi^{(p+1)}\).

\[
\pi^{(p+1)} = \frac{x_2^{(p)} + 34}{x_2^{(p)} + 34 + 18 + 20}
\]
We use \((x_1^{(p)}, x_2^{(p)}, 18, 20, 34)\) as if these estimated data were the observed data, and find the maximum likelihood estimate of \(\pi\), denoted \(\pi^{(p+1)}\).

\[
\pi^{(p+1)} = \frac{x_2^{(p)} + 34}{x_2^{(p)} + 34 + 18 + 20}
\]

And we go back to the E-step to complete the \((p + 1)\)-th iteration.
We start with $\pi^{(0)} = 0.5$, and the algorithm converges in eight steps:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\pi^{(p)}$</th>
<th>$\pi^{(p)} - \pi^*$</th>
<th>$\frac{(\pi^{(p+1)} - \pi^<em>)}{(\pi^{(p)} - \pi^</em>)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5000000000</td>
<td>0.126821498</td>
<td>0.1465</td>
</tr>
<tr>
<td>1</td>
<td>0.608247423</td>
<td>0.018574075</td>
<td>0.1346</td>
</tr>
<tr>
<td>2</td>
<td>0.624321051</td>
<td>0.002500447</td>
<td>0.1330</td>
</tr>
<tr>
<td>3</td>
<td>0.626488879</td>
<td>0.000332619</td>
<td>0.1328</td>
</tr>
<tr>
<td>4</td>
<td>0.626777323</td>
<td>0.000044176</td>
<td>0.1328</td>
</tr>
<tr>
<td>5</td>
<td>0.626815632</td>
<td>0.000005866</td>
<td>0.1328</td>
</tr>
<tr>
<td>6</td>
<td>0.626820719</td>
<td>0.00000779</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.626821395</td>
<td>0.00000104</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.626821484</td>
<td>0.00000014</td>
<td></td>
</tr>
</tbody>
</table>

At each step we use $\pi^{(p)} = \pi^*$ and $\pi^{(p+1)} = \pi^*$ to solve for $\pi^*$ as the maximum-likelihood estimate of $\pi$. 
Applications of EM algorithm

- Missing Data
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling
- Grouping
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling

- Grouping

- Censoring and Truncation
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling
- Grouping
- Censoring and Truncation
- Finite Mixtures
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling
- Grouping
- Censoring and Truncation
- Finite Mixtures
- Hyperparameter Estimation
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling
- Grouping
- Censoring and Truncation
- Finite Mixtures
- Hyperparameter Estimation
- Iteratively Reweighted Least Squares
Applications of EM algorithm

- Missing Data
  - Multinomial sampling
  - Normal linear model
  - Multivariate normal sampling

- Grouping
- Censoring and Truncation
- Finite Mixtures
- Hyperparameter Estimation
- Iteratively Reweighted Least Squares
- Factor Analysis
Example: Old Faithful

Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming.
$X = \text{Waiting time between eruptions.}$

$p = \text{Probability that eruption is of a shorter waiting time}$

$\theta = (p, \mu_1, \mu_2, \sigma_1, \sigma_2)$

$$f_X(x|\theta) = pN(\mu_1, \sigma_1) + (1 - p)N(\mu_2, \sigma_2)$$

Define:

$$Y_i = \begin{cases} 
1 & X_i \text{ has shorter waiting time} \\
0 & X_i \text{ has longer waiting time} 
\end{cases}$$

$Y_i \sim \text{Bern}(p)$ and $Y_i$ is missing data
Old Faithful: E step

\[ Y_i | X_i, \theta^{(k)} \sim \text{Bin}(1, p_i^{(k)}) \]

where

\[ p_i^{(k)} = \frac{p^{(k)} N(\mu_1^{(k)}, \sigma_1^{(k)})}{p^{(k)} N(\mu_1^{(k)}, \sigma_1^{(k)}) + (1 - p^{(k)}) N(\mu_2^{(k)}, \sigma_2^{(k)})} \quad \text{at } X_i \]

Thus,

\[ E(Y_i | X_i, \theta^{(k)}) = p_i^{(k)} \]
Old Faithful: M step

\[ L(\theta|X, Y) = \prod_{i=1}^{n} p^{Y_i} [N(\mu_1, \sigma_1)]^{Y_i} (1 - p)^{1-Y_i} [N(\mu_2, \sigma_2)]^{1-Y_i} \]

Take log and replace \( Y_i \) with \( p^{(k)}_i \), then maximize for \( \theta \).

\[
p^{(k+1)} = \frac{1}{n} \sum_{i=1}^{n} p^{(k)}_i
\]

\[
\mu^{(k+1)}_1 = \frac{\sum_{i=1}^{n} p^{(k)}_i X_i}{\sum_{i=1}^{n} p^{(k)}_i}
\]

\[
\mu^{(k+1)}_2 = \frac{\sum_{i=1}^{n} (1 - p^{(k)}_i) X_i}{\sum_{i=1}^{n} (1 - p^{(k)}_i)}
\]

\[
\sigma^{(k+1)}_1 = \frac{\sum_{i=1}^{n} p^{(k)}_i (X_i - \mu^{(k+1)}_1)^2}{\sum_{i=1}^{n} p^{(k)}_i}
\]

\[
\sigma^{(k+1)}_2 = \frac{\sum_{i=1}^{n} (1 - p^{(k)}_i) (X_i - \mu^{(k+1)}_1)^2}{\sum_{i=1}^{n} (1 - p^{(k)}_i)}
\]
$p^{(0)} = 0.5$, $\mu_1^{(0)} = 52$, $\mu_2^{(0)} = 82$, $\sigma_1^{(0)} = 4$, $\sigma_2^{(0)} = 4$
Estimates

em <- function(W,s){

Ep <- s[1]*dnorm(W, s[2], sqrt(s[4]))/
(s[1]*dnorm(W, s[2], sqrt(s[4]))+ 
(1-s[1])*dnorm(W, s[3], sqrt(s[5])))

s[1] <- mean(Ep)

s[2] <- sum(Ep*W) / sum(Ep)


s[4] <- sum(Ep*(W-s[2])^2) / sum(Ep)

s[5] <- sum((1-Ep)*(W-s[3])^2) / sum(1-Ep)

s}

Iterations

iter <- function(W, s){

s1 <- em(W,s)

cutoff <- rep(.0001,5)

if(sum(s-s1>cutoff) > 0){

s = s1

iter(W,s)

}

else s1

}

Implementation

> W <- faithful$waiting
> s <- c(0.5, 52, 82, 16, 16)
> iter(W,s)

[1]  0.3608866  54.6148747  80.0910812  34.4714038  34.4301694
Estimated Distribution

Histogram

Density

Waiting Time

40 50 60 70 80 90 100
Multiple Imputation Overview

- Imputation is ‘filling in’ missing data with plausible values

- Rubin (1987) conceived a method, known as multiple imputation, for valid inferences using the imputed data
  - Multiple Imputation is a Monte Carlo method where missing values are imputed \( m > 1 \) separate times (typically \( 3 \leq m \leq 10 \))

- Multiple Imputation is a three step procedure:
  - **Imputation**: Impute the missing entries in the data \( m \) separate times
  - **Analysis**: Analyze each of the \( m \) complete data sets separately
  - **Pooling**: Combine the \( m \) analysis results into a final result
**Theory**

- $Q$ is some statistic of scientific interest in the population
  - Could be population means, regression coefficients, population variances, etc.
  - $Q$ cannot depend on the particular sample

- We estimate $Q$ by $\hat{Q}$ or $\bar{Q}$ along with a valid estimate of its uncertainty
  - $\hat{Q}$ is the estimate from complete data
  - $\hat{Q}$ accounts for sampling uncertainty
  - $\bar{Q}$ is a pooled estimate
  - $\bar{Q}$ accounts for sampling and missing data uncertainty
\( \hat{Q} \) and \( \bar{Q} \)

- \( \hat{Q}_i \) is our estimate from the \( i \)-th imputation
  - \( \hat{Q}_i \) has \( k \) parameters
  - \( \hat{Q}_i \) \( k \times 1 \) column vector

- To compute \( \bar{Q} \) we simply average over all \( m \) imputations

\[
\bar{Q} = \frac{1}{m} \sum_{i=1}^{m} \hat{Q}_i
\]
Let $U$ be the squared standard error of $Q$

We estimate $U$ by $\bar{U}$

- $\hat{U}_i$ is the covariance matrix of $\hat{Q}_i$, our estimate from the $i$-th imputation

$$\bar{U} = \frac{1}{m} \sum_{i=1}^{m} \hat{U}_i$$

Notice: $\hat{U}_i$ is the variance within the estimate $\hat{Q}_i$

Let $B$ be the variance between the $m$ complete-data estimates:

$$B = \frac{1}{m-1} \sum_{i=1}^{m} (\hat{Q}_i - \bar{Q})(\hat{Q}_i - \bar{Q})^\top$$
Let $T$ denote the total variance of $\tilde{Q}$

- $T \neq \bar{U} + B$

$T$ is computed by:

$$T = \bar{U} + B + \frac{B}{m}$$

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

where $\frac{B}{m}$ is simulation error.
Summary

\[ T = \bar{U} + (1 + \frac{1}{m})B \]

The intuition for $T$ is as follows:
- $\bar{U}$ is the variance in $\bar{Q}$ caused by the fact that we are using a sample.
- $B$ is the variance caused by the fact that there were missing values in our sample.
- $\frac{B}{m}$ is the simulation variance from the fact that $\bar{Q}$ is based on a finite $m$. 
Tests and Confidence Intervals

- For multiple imputation to be valid, we must first assume, that with complete data
  \[ (\hat{Q} - Q)/\sqrt{U} \sim \mathcal{N}(0, 1) \]
  would be appropriate

- Then, after our multiple imputation steps, tests and confidence intervals are based on a Student's t-approximation
  \[ (\bar{Q} - Q)/\sqrt{T} \sim t_v \]
  \[ v = (m - 1) \left[ 1 + \frac{\bar{U}}{(1 + \frac{1}{m})B} \right]^2 \]
**Imputation Step**

- The validity of inference relies on how imputations are generated.

- Rubin proposed three conditions under which multiple imputation inference is "randomization-valid"

\[
E(\bar{Q}|Y) = \hat{Q} \quad (1) \\
E(\bar{U}|Y) = U \quad (2) \\
(1 + \frac{1}{m})E(B|Y) \geq V(\bar{Q}) \quad (3)
\]

- **Result**: If the complete-data inference is randomization valid and the our imputation procedure satisfies the proceeding conditions, then our finite \( m \) multiple imputation inference is also randomization-valid.

  - Not always easy to get these conditions, often requires Bayesian approach
The mice package does multiple imputation in R

```r
> library(mice)
> head(nhanes)

    age  bmi  hyp  chl
   1   1   NA  NA
   2   2 22.7   1 187
   3   1   NA   1 187
   4   3   NA  NA
   5   1 20.4   1 113
   6   3   NA  NA
```

We're interested in the simple linear regression of BMI on Age

- $Q = \beta_1$ from $E(BMI|Age) = \beta_0 + Age^\top \beta_1$
The *mice* package has some nice functions that summarize our missing data.

```
> md.pattern(nhanes)
   age hyp bmi chl
13  1   1   1   1   0
  1   1   1   0   1   1
  3   1   1   1   0   1
  1   1   0   0   1   2
  7   1   0   0   0   3
   0   8   9  10  27
```

Above, the output shows we have 13 complete rows, 1 missing only BMI, 3 missing Cholesterol, 1 missing Hypertension and BMI, and 7 missing Hypertension, BMI, and Cholesterol.
Simple Example in R

```r
> library(VIM)
> marginplot(nhanes[c(1,2)], col = c("blue", "red", "orange"))
```
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Scale type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmm</td>
<td>Predictive mean matching</td>
<td>numeric</td>
</tr>
<tr>
<td>norm</td>
<td>Bayesian linear regression</td>
<td>numeric</td>
</tr>
<tr>
<td>norm.nob</td>
<td>Linear regression, non-Bayesian</td>
<td>numeric</td>
</tr>
<tr>
<td>norm.boot</td>
<td>Linear regression with bootstrap</td>
<td>numeric</td>
</tr>
<tr>
<td>mean</td>
<td>Unconditional mean imputation</td>
<td>numeric</td>
</tr>
<tr>
<td>2L.norm</td>
<td>Two-level linear model</td>
<td>numeric</td>
</tr>
<tr>
<td>logreg</td>
<td>Logistic regression</td>
<td>factor, 2 levels</td>
</tr>
<tr>
<td>logreg.boot</td>
<td>Logistic regression with bootstrap</td>
<td>factor, 2 level</td>
</tr>
<tr>
<td>polyreg</td>
<td>Multinomial logit model</td>
<td>factor &gt; 2 levels</td>
</tr>
<tr>
<td>polr</td>
<td>Ordered logit model</td>
<td>ordered, &gt; 2 levels</td>
</tr>
<tr>
<td>lda</td>
<td>Linear discriminant analysis</td>
<td>factor</td>
</tr>
<tr>
<td>sample</td>
<td>Simple random sample</td>
<td>any</td>
</tr>
</tbody>
</table>
Imputation Approaches

- Except in trivial settings, the probability distributions that we draw from to give 'proper' multiple imputation tend to be complicated
  - Often requires MCMC

- In our example, we will use an approach called Predictive Mean Matching
  - Calculate $\hat{Y}_{observed} = \{\hat{y}_i = x_i^T \beta : i \in Observed\}$
  - For $y_{missing}$, calculate $\hat{Y}_{missing} = \{\hat{y}_j = x_i^T \beta : j \in Missing, i \in Observed\}$
  - Among our $\hat{Y}_{observed}$, locate the observation whose predicted value is closest to $\hat{y}_j$ for all $j \in Missing$ and impute that value
  - For $m = n$, impute random draws the from the $n$ observations whose predicted value is closest to $\hat{y}_m$
Predictive Mean Matching
We use the *mice()* function to run multiple imputation using predictive mean modeling

```r
imp.nhanes <- mice(nhanes, m=5, method="pmm", print=FALSE, seed=8053)
```

We can look at our imputed values for BMI and notice these are sampled observed values

```r
imp.nhanes$imp$bmi

1   2   3   4   5
1 22.5 25.5 27.2 22.0 33.2
3 26.3 30.1 30.1 35.3 33.2
16 22.5 25.5 29.6 30.1 28.7
21 25.5 35.3 27.5 30.1 35.3

na.omit(nhanes$bmi)

[1] 22.7 20.4 22.5 30.1 22.0 21.7 28.7 29.6 27.2 26.3
[11] 35.3 25.5 33.2 27.5 24.9 27.4
```
We fit five separate linear regression models

```r
> fit <- with(imp.nhanes, lm(bmi ~ age))
```

We average our estimates using `pool()` from the `mice` package

```r
> est <- pool(fit)
> est$qbar
   (Intercept) age
30.24  -2.06
```
Using the *mice()* package, we can make valid inferences

```r
> summary(est)

                 est        se       t       df
(Intercept) 30.242705 2.944000 10.272659 4.719653
age           -2.060628 1.288428 -1.599336 7.255069

                       Pr(>|t|)        lo 95      hi 95   nmis
(Intercept) 0.0002086732 22.537686 37.9477244   NA
age            0.1522742652 -5.085695  0.9644395   0

             fmi  lambda
(Intercept) 0.7087166 0.6068631
age           0.5605660 0.4541020
```

\[ p \approx .15 \implies \text{no age effect} \]
Questions?