Mu Sigma

Handling Missing Data

Mu Sigma University

Do The Math

Chicago, IL
Bangalore, India
www.mu-sigma.com

May 2016

Agenda

- Various Aspects of Missing Data
- Missing Value Strategies
- Case-wise Deletion
- Paire-wise Deletion
- Mean Imputation
- Hot Deck Imputation
- Regression Imputation
- k-Nearest Neighbour Imputation
- Model-Based Imputation: Maximum Likelihood by EM Algorithm
- Multiple Imputation
- Little's Test for MCAR

Missing Values are Common

- In univariate data, some values are missing
- In multivariate data
  - the entire data for a case is missing
  - data on one or more variables are missing
- Innumerable reasons for "missingness"
- Do missing data affect analysis or need different analysis? YES; YES
- Is there need to "impute" missing data?
- What are the different methods of imputation?

Types of Missing Values

Take the example of scores in various tests of a class of students. Data may be missing for one or more tests of one or more students due to absence in test

- **Missing completely at random (MCAR)**: data are missing independently of both observed and unobserved data.
  Example: a student is absent due to accident on way to school
- **Missing at random (MAR)**: given the observed data, data are missing independently of unobserved data
  Example: Student's absence is not related to a possible performance in the missed test—presumably a good student—absence is due to death in family
- **Missing Not at Random (MNAR)**: missing observations related to values of unobserved data
  Example: A student is absent in a test because he is a bad student and/or they have not prepared well

Consequences of Missing Data

- MCAR implies MAR, but not the other way around
- Most methods assume MAR
- We can ignore missing data, that is, analyze without the missing cases, if we have MAR or MCAR
- Informative missingness: the fact that data is missing contains information about the response
- Observed data is biased sample. Missing data cannot be ignored
- Cannot distinguish MAR from MNAR without additional information
- With MNAR, you get a non-representative sample and biased estimates

Definitions of Missing Data Mechanisms

- Definition: Y_{com} is the complete data, which consists of Y_{obs}, the observed part, and Y_{mis}, the missing part: $Y_{com} = (Y_{obs}; Y_{mis})$
- Missing data is
  - Missing at Random (MAR): $P(R|Y_{com}) = P(R|Y_{obs})$
    missingness is not related to missing scores
  - Missing Completely at Random (MCAR): $P(R|Y_{com}) = P(R)$
    missingness is not related to observed or missing scores
  - Missing Not at Random (MNAR): missingness is related to missing scores Y_{mis} (and observed)

Why are Missing Data a Problem?

- Missing data destroys the balance and symmetry in data
- Data set is not an $n \times p$ matrix, n: number of cases; p number of variables
- Most data analysis procedures and statistical software were designed for a full $n \times p$ data matrix and not designed to handle missing data
- or handle missing data in an ad hoc manner
- The assumptions of random samples and data models are violated
- Ignoring missing data or (ad hoc) editing lend an appearance of completeness, but may lead to serious problems
- inefficiency (loss of information) leads to loss of power
- systematic differences leads to biased results
- and unreliable results

Impossibility of Showing Missingness is Random

- Missingness at random (MAR) is relatively easy to handle
- Unfortunately we cannot be sure whether data really are MAR or whether the missingness depends on unobserved predictors or the missing data themselves
- We generally must make assumptions, or check with reference to other studies (for example, surveys in which extensive follow-ups are done in order to ascertain the earnings of nonrespondents)
- In practice, we typically try to include as many predictors as possible in a model so that the "missing at random" assumption is reasonable
- Many missing data approaches simplify the problem by throwing away data
- These approaches may lead to biased estimates

Ignorable and Nonignorable Missingness

- When data that are MNAR (Missing Not At Random), life becomes very much more difficult
- We need to understand and model the mechanism that causes missingness
- Modeling missingness is a difficult exercise
- On the other hand, if data are at least MAR, the mechanism for missingness is ignorable
- Thus we can proceed without worrying about the model for missingness
- This is not to say that we can just ignore the problem of missing data
- We still want to find better estimators of the parameters in our model, but we don't have to write a model for missingness
- We certainly have enough to do to improve estimation without also worrying about why the data are missing

Missing Value Analysis Strategies

- Deletion Methods
  - Listwise deletion
  - Pairwise deletion
- Single Imputation Methods
  - Mean/mode substitution
  - Regression Imputation
  - Hot Deck Imputation
  - k-Nearest Neighbor Imputation
- Model-Based Methods
  - Maximum Likelihood
  - Multiple imputation

No Treatment Option: Complete Case Method

- The simplest way to deal with missing data is to use only those cases with no missing values in their data or have no missing data on the variables we want to analyze
- This approach is called "complete case analysis" or "listwise deletion"
- If the missing data are MCAR, this approach provides unbiased estimates, though estimates are not statistically efficient
- If the missing data are not MCAR, this approach will result in biased estimates
- A large number of cases may be thrown away resulting in a huge reduction in sample size and in estimation efficiency

Complete-Case Method: An Example: Data Set: salespop.txt

This data set consists of sales (sales) of stores (in $'s K) in a week, the store area (area) (in 000 sq. ft.), the town population (tpop) (in '000s) and the neighborhood population (npop) (in 000's)

A few rows of data are given below (NA indicating missing data):

tpop npop area sales

```
16  NA  50 15
27 6 67 93
94 26 291 637
96 48 331 1263
203 99 599 2796
314 152 925 4214
395 207 NA  5419
445 233 1338 6559
473 271 1641 7536
467 268 1564 8136
438 236 1527 7023
477 NA 1548 7524
440 239 1431 6660
355 179 1213 5293
283 115 911 3468
```

Complete-Case Method: An Example Continued: R Code

```
> salespop<-read.table("C://documents and settings//krishnan.t
+                      desktop//salespop.txt",header=TRUE)
> salespoplm<-lm(sales~.,data=salespop)
> summary(salespoplm)
```

Complete-Case Method: An Example Continued: R Output

```
Call:
lm(formula = sales ~ ., data = salespop)

Residuals:
     Min       1Q   Median       3Q      Max
-1255.62   -85.36   -17.08    75.80   926.28

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.7983    21.2981   0.695  0.48760
tpop          0.1059     0.7988   0.133  0.89458
npop         31.8576     0.4689  67.935  < 2e-16 ***
area         -0.6398     0.2446  -2.615  0.00928 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

R Output Continued

```
Residual standard error: 248.2 on 374 degrees of freedom
  (6 observations deleted due to missingness)
Multiple R-squared: 0.992,  Adjusted R-squared: 0.9919
F-statistic: 1.541e+04 on 3 and 374 DF,  p-value: < 2.2e-16
```

- Note that 6 of the 384 original observations are deleted due to missingness
- These cases, however, are partially observed, and contain valuable information about the relationships between those variables which are present in the partially completed observations
- Multiple imputation will help us retrieve that information and help make better, more efficient, inferences

No Treatment Option: Available Case Method

- The “available case method” or “pairwise deletion” is another way of not treating the missing data
- The pairwise deletion methods use only the data that are available
- For example, in computing the correlation coefficient between a pair of variables, we use only the cases (pairs) that have non-missing responses on both of the variables
- In general, pairwise deletion is less preferred than the listwise deletion method

Available Case Method: An Example: R Code

We compute the covariance matrix of salespop variables without specifying how to deal with missing cases and specifying to compute it only with available cases

```
> salespop<-read.table("C://documents and settings//
+     krishnan.t//desktop//salespop.txt",header=TRUE)
> cov(salespop)
> cov(salespop,use="complete.obs")
```

Available Case Method: An Example: R Output

```
           tpop npop area     sales
tpop   27468.65   NA   NA  432617.1
npop         NA   NA   NA        NA
area         NA   NA   NA        NA
sales 432617.08   NA   NA 7661582.4

           tpop       npop       area   sales
tpop   27334.40  15247.076   91888.46  429845
npop   15247.08   9302.406   51655.12  264921
area   91888.46  51655.125  311827.18 1455850
sales 429845.02 264921.031 1455850.21 7615013
```

Problems with Pairwise Deletion

- In pairwise deletion, each computed statistic may be based on a different subset of cases
- This can be problematic
- For example, a covariance or correlation matrix computed using pairwise deletion may not be positive semidefinite. That is, it may have negative eigenvalues (negative variances for some derived variables!), which can create problems for various statistical analyses like regression, factor analysis, etc.
- This can occur because when correlations are computed using different cases, the resulting patterns can be ones that are impossible to produce with complete data
- This may result in variances of some derived variables to be negative and correlations between derived variables to be outside the (-1,+1) range
- But the greater danger is incorrect statistics even if they are within range

Mean Substitution Method

- For all cases that have a missing value for the variable under consideration, the mean substitution method substitutes the computed available cases mean
- This procedure is simple to implement but has the following disadvantages:
  - It distorts the underlying distribution of the data, making the distribution more peaked around the mean and reducing the variance in the variable
  - It does not take into account the fact that the imputed data have more uncertainty than does a complete data set
  - Although this method is slightly better than the available case method, it still will lead to biased results and thus is generally not recommended
  - This method will yield biased estimates regardless of the type of "missingness"
  - Sometimes, especially if the distribution is skewed, the median is substituted rather than the mean

Mean Substitution Example: R Code

```
> salespop<-read.table("C://documents and settings//
+     krishnan.t//desktop//salespop.txt",header=TRUE)
> require(HotDeckImputation)
> salespop1<-as.matrix(salespop)
> impute.mean(data=salespop1)
```

Mean Substitution Example: R Output

	[,1]	[,2]	[,3]	[,4]
[1,]	59	15.00000	178.0000	313
[2,]	31	6.00000	146.0000	105
[3,]	30	4.00000	106.0000	59
[4,]	20	1.00000	60.0000	19
[5,]	7	1.00000	16.0000	24
[6,]	16	96.75853	50.0000	15
[7,]	27	6.00000	67.0000	93
[8,]	94	26.00000	291.0000	637
[9,]	96	48.00000	331.0000	1263
[10,]	203	99.00000	599.0000	2796
[11,]	314	152.00000	925.0000	4214
[12,]	395	207.00000	662.5564	5419
[13,]	445	233.00000	1338.0000	6559
[14,]	473	271.00000	1641.0000	7536
[15,]	467	268.00000	1564.0000	8136
[16,]	438	236.00000	1527.0000	7023
[17,]	477	96.75853	1548.0000	7524
[18,]	440	239.00000	1431.0000	6660
[19,]	355	179.00000	1213.0000	5293
[20,]	283	115.00000	911.0000	3468
[21,]	231	93.00000	810.0000	3007
[22,]	194	62.00000	610.0000	1868
[23,]	131	33.00000	393.0000	812
[24,]	68	14.00000	245.0000	453
[25,]	48	7.00000	151.0000	145
[26,]	32	5.00000	662.5564	128
[27,]	31	5.00000	90.0000	126
[28,]	13	2.00000	37.0000	55
[29,]	10	2.00000	28.0000	51
[30,]	19	3.00000	46.0000	57

More on Mean Substitution

- There are a few problems with this approach
- It adds no new information
- The overall mean, with or without replacing any missing data, will be the same
- In addition, such a process leads to an underestimate of error
- We have really added no new information to the data but we have increased the sample size
- The effect of increasing the sample size is to increase the denominator for computing the standard error, thus reducing the standard error
- Adding no new information certainly should not make you more comfortable with the result, but this would seem to suggest just that

Imputation

Imputation: missing data points in a dataset are replaced with plausible values

- **Mean imputation**: missing data points are simply replaced with the mean
- **Random imputation**: missing data points are imputed randomly from a random uniform distribution
- **Regression-based imputation**: missing values are replaced by a predicted score generated by a regression model based on the non-missing data

Hot Deck Imputation

- This method sorts respondents and non-respondents into a number of imputation subsets according to a user-specified set of covariates
- An imputation subset comprises cases with the same values as those of the user-specified covariates
- Missing values are then replaced with values taken from matching respondents (i.e. respondents that are similar with respect to the covariates)
- If there is more than one matching respondent for any particular non-respondent, the user has two choices:
  - The first respondent's value as counted from the missing entry downwards within the imputation subset is used to impute. The reason for this is that the first respondent's value may be closer in time to the case that has the missing value. For example, if cases are entered according to the order in which they occur, there may possibly be some type of time effect in some studies
  - A respondent's value is randomly selected from within the imputation subset. If a matching respondent does not exist in the initial imputation class, the subset will be collapsed by one level starting with the last variable that was selected as a sort variable, or until a match can be found. Note that if no matching respondent is found, even after all of the sort variables have been collapsed, three options are available as follows:

Three Options if Respondents Do Not Match

- Re-specify new sort variables
- Perform random overall imputation where the missing value will be replaced with a value randomly selected from the observed values in that variable
- Do not impute the missing value

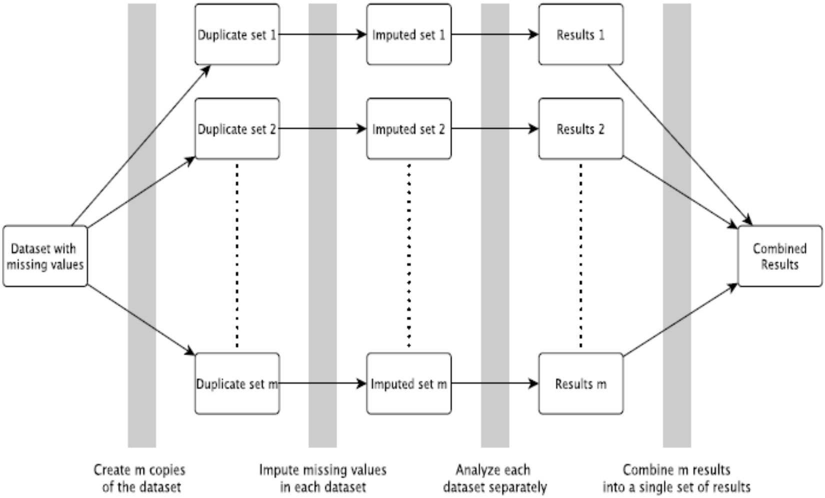
Hot Deck Imputation Algorithm

From: We Have to Be Discrete About This: A Non-Parametric Imputation Technique for Missing Categorical Data. Skyler J. Cranmer and Jeff Gill, April 30, 2012, British Journal of Political Science

THE MULTIPLE HOT DECK IMPUTATION ALGORITHM

1. Create several copies of the dataset.
2. Search down columns of the data sequentially looking for missing observations.
   a. When a missing value is found, compute a vector of affinity scores, for that missing value.
   b. Create the cell of best donors and draw randomly from it to produce a vector of imputations.
   c. Impute one of these values into the appropriate cell of each duplicate dataset.
4. Repeat Step 2 until no missing observations remain.
5. Estimate the statistic of interest for each dataset.
6. Combine the estimates of the statistic into a single estimate.

Hot Deck Multiple Imputation



Example: Data Description: Data File: fitness.txt

- Measurements were made on men involved in a physical fitness course
- Variables are:
  - Oxygen (intake rate, ml per kg body weight per minute)
  - Runtime (time to run 1.5 miles in minutes)
  - RunPulse (heart rate while running)
- Certain values are missing (denoted by .)

The Data

Oxygen	RunTime	RunPulse
44.609	11.37	178
45.313	10.07	185
54.297	8.65	156
59.571	.	.
49.874	9.22	.
44.811	11.63	176
45.681	11.95	176
49.091	10.85	.
39.442	13.08	174
60.055	8.63	170
50.541	.	.
37.388	14.03	186
44.754	11.12	176
47.273	.	.
51.855	10.33	166
49.156	8.95	180
40.836	10.95	168
46.672	10.00	.
46.774	10.25	.
50.388	10.08	168
39.407	12.63	174
46.080	11.17	156
45.441	9.63	164
54.625	8.92	146
45.118	11.08	.
39.203	12.88	168
45.790	10.47	186
50.545	9.93	148
48.673	9.40	186
47.920	11.50	170
47.467	10.50	170

Hot Deck Imputation Example: R Code

```
> fitness<-read.table("C://documents and settings//krishnan.t/
> require(HotDeckImputation)
> fitness1<-as.matrix(fitness)
> impute.NN_HD(data=fitness1,distance="man")
```

Hot Deck Imputation Example: R Output

```
        [,1]  [,2] [,3]
 [1,] 44.609 11.37  178
 [2,] 45.313 10.07  185
 [3,] 54.297  8.65  156
 [4,] 59.571  8.63  170
 [5,] 49.874  9.22  180
 [6,] 44.811 11.63  176
 [7,] 45.681 11.95  176
 [8,] 49.091 10.85  170
 [9,] 39.442 13.08  174
[10,] 60.055  8.63  170
[11,] 50.541  9.93  148
[12,] 37.388 14.03  186
[13,] 44.754 11.12  176
[14,] 47.273 10.50  170
[15,] 51.855 10.33  166
[16,] 49.156  8.95  180
[17,] 40.836 10.95  168
[18,] 46.672 10.00  185
[19,] 46.774 10.25  170
[20,] 50.388 10.08  168
[21,] 39.407 12.63  174
[22,] 46.080 11.17  156
[23,] 45.441  9.63  164
[24,] 54.625  8.92  146
[25,] 45.118 11.08  176
[26,] 39.203 12.88  168
[27,] 45.790 10.47  186
[28,] 50.545  9.93  148
[29,] 48.673  9.40  186
[30,] 47.920 11.50  170
```

Regression Substitution Method

- We use the complete data points to calculate the regression of the incomplete variable on the other complete variables
- Then we substitute the predicted mean for each unit with a missing value
- In this way we use information from the joint distribution of the variables to make the imputation
- Regression mean imputation can generate unbiased estimates of means, associations and regression coefficients in a much wider range of settings than simple mean imputation
- However, one important problem remains. The variability of the imputations is too small, so the estimated precision of regression coefficients will be wrong and inferences will be misleading

Regression Imputation: An Example

- We consider the *fitness* data set.
- The variable Oxygen has complete data
- The variable RunTime has three observations missing
- The variable RunPulse has three observations (4, 11, 14) missing together with RunTime and five on its own (5, 8, 18, 19, 25)
- So we develop three regression lines as follows:
  - RunTime on Oxygen to predict missing observations 4, 11, 14
  - RunPulse on Oxygen to predict missing observations 4, 11, 14
  - RunPulse on Oxygen and RunTime to predict missing observations 5, 8, 18, 19, 25

Regression Imputation Example: R Code

```
> fitness<-read.table("C://documents and settings//krishnan.t/
> x2onx1<-lm(RunTime~Oxygen,data=fitness)
> new<-data.frame(Oxygen=c(59.571,50.541,47.273))
> predict(x2onx1,new)
> x3onx1<-lm(RunPulse~Oxygen,data=fitness)
> predict(x3onx1,new)
> x3onx1x2<-lm(RunPulse~RunTime+Oxygen,data=fitness)
> new2<-data.frame(Oxygen=c(49.874,49.091,46.672,46.774,45.118
> predict(x3onx1x2,new2)
```

Regression Imputation Example: R Output

```
        1         2         3
 7.733491  9.827755 10.585679

       1        2        3
159.1726 167.2775 170.2106

       1        2        3        4        5
168.7270 167.9277 171.5581 171.1831 172.2053
```

Completed Data Set

predictOxygen	RunTime	RunPulse
44.609	11.37	178
45.313	10.07	185
54.297	8.65	156
59.571	7.73	159
49.874	9.22	169
44.811	11.63	176
45.681	11.95	176
49.091	10.85	168
39.442	13.08	174
60.055	8.63	170
50.541	9.82	167
37.388	14.03	186
44.754	11.12	176
47.273	10.59	17
51.855	10.33	166
49.156	8.95	180
40.836	10.95	168
46.672	10.00	172
46.774	10.25	171
50.388	10.08	168
39.407	12.63	174
46.080	11.17	156
45.441	9.63	164
54.625	8.92	146

k-Nearest Neighbor Approach

- Another way of dealing with missing data is the k nearest neighbor (knn) approach
- This method is quite simple in principle but is effective and often preferred over some of the more sophisticated methods described above
- Nearest neighbors are records that have similar completed data patterns; the average of the k-nearest neighbors' completed data are used to impute the value for a variable that is missing its value
- k can be set by the analyst
- It has been shown that a k ranging from 5 to 10 is adequate
- The advantage of the knn approach is that it assumes data are missing at random (MAR) meaning, missing data only depends on the observed data; which in turn means, the knn approach is able to take advantage of multivariate relationships in the completed data
- The disadvantage of this approach is it does not include a component to model random variation; consequently uncertainty in the imputed value is underestimated

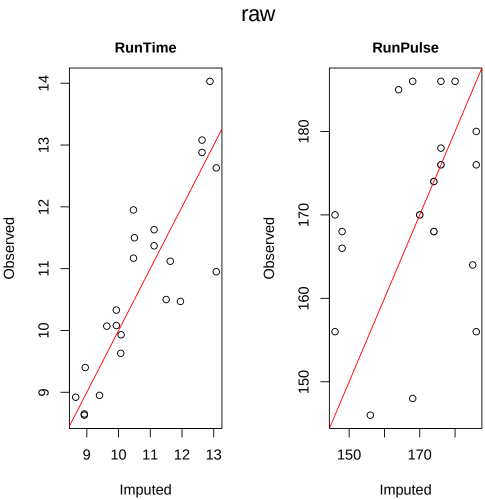
k-Nearest Neighbor Approach: Example: R Code

```
> library(yaImpute)
> data(fitness)
> set.seed(1)
> refs=sample(rownames(fitness),
+       c(1,2,3,6,7,9,10,12,13,15,16,17,20:24))
> x <- as.matrix(fitness[, 1])
> y <- fitness[, 2:3]
> raw <- yai(x = x, y = y, method = "euclidean")
> plot(raw)
> tail(impute(raw))
```

k-Nearest Neighbor Approach: Example: R Output

```
   RunTime RunPulse RunTime.o RunPulse.o
8     8.95      180        NA         NA
11    9.93      148        NA         NA
14   10.50      170        NA         NA
18   11.17      156        NA         NA
19   10.50      170        NA         NA
25   10.07      185        NA         NA
```

k-Nearest Neighbor Approach: Example: R Output: Graphs of Imputed Values



k-Nearest Neighbor Approach: Example: R Output Missing Value Estimates

	RunTime	RunPulse	RunTime.o	RunPulse.o
1	11.12	176	11.37	178
2	9.63	164	10.07	185
3	8.92	146	8.65	156
6	11.12	176	11.63	176
7	10.47	186	11.95	176
9	12.63	174	13.08	174

Maximum Likelihood Estimation

- For many analyses like Principal Component Analysis, Structural Equation Modeling, an estimate of the mean vector and covariance matrix are needed
- The likelihood can be written of the available variables and maximized
- This is very different from the previous methods
- The previous methods were concerned with retrieving a new (imputed) data file
- The maximum likelihood method (implemented in R by the package mvnmle) is concerned only with a complete variance/covariance matrix based on maximum likelihood values from the available data
- This maximum likelihood estimation is a computer-intensive iterative method

mvnmle Package in R

- We do this computation on the "fitness" data
- If you ask for a covariance matrix the output has NA if data are missing
- But proper maximum likelihood estimates can be computed

```
library(mvnmle)
cov(fitness)
           Oxygen RunTime RunPulse
Oxygen   28.37938      NA       NA
RunTime        NA      NA       NA
RunPulse       NA      NA       NA

mlest(fitness)
$muhat
[1]  47.37579  10.56183 170.17586

$sigmahat
           [,1]      [,2]       [,3]
[1,]  27.463985 -6.369522 -24.615723
[2,]  -6.369522  1.998728   5.168555
[3,] -24.615723  5.168555 120.441842
```

EM Algorithm

- The two most important treatments of missing data are expectation/maximization (known as the EM algorithm) and multiple imputation (MI)
- These are not distinct models, and EM is often used as a starting point for MI
- EM is a maximum likelihood procedure that works with the relationship between the unknown parameters of the data model and the missing data
- If we knew the missing values, then estimating the model parameters would be straightforward
- Similarly, if we knew the parameters of the data model, then it would be possible to obtain unbiased predictions for the missing values
- This suggests an approach in which we first estimate the parameters, then estimate the missing values, then use the filled in data set to re-estimate the parameters, then use the re-estimated parameters to estimate missing values, and so on
- When the process finally converges on stable estimates we stop iterating

EM Algorithm Details for Multivariate Normal

- Let us assume a multivariate normal model
- Suppose that we have a data set with five variables (X_1 to X_5), with missing data on one or more variables
- The algorithm first performs a straightforward regression imputation procedure where it imputes values of X_1, for example, from the other four variables, using the parameter estimates of means, variances, and covariances or correlations from the available data
- After imputing data for every missing observation in the data set, EM calculates a new set of parameter estimates
- The estimated means are simply the means of the variables in the imputed data set
- EM corrects biased estimation by estimating variances and covariances that incorporate the residual variance from the regression
- Now that we have a new set of parameter estimates, we repeat the imputation process to produce another set of data
- From that new set we re-estimate our parameters, as above, and then impute yet another set of data
- This process continues in an iterative fashion until the estimates converge

Single and Multiple Imputation

- A problem with imputing only a single value for every missing value is that this does not reflect our uncertainty about the predictions
- Standard errors may therefore be biased (too small)
- An alternative is to replace each missing value with multiple plausible values
- This represents the uncertainty about the right value to impute
- Data analyses from multiply-imputed datasets can be combined to produce estimates and confidence intervals that incorporate missing-data uncertainty

Multiple Imputation

- An additional method for imputing values for missing observations is known as multiple imputation (MI)
- There are a number of ways of performing MI, though they all involve the use of random components to overcome the problem of underestimation of standard errors
- The parameter estimates using this approach are nearly unbiased
- The interesting thing about MI is that the word "multiple" refers not to the iterative nature of the process involved in imputation, but to the fact that we impute multiple complete data sets and run whatever analysis is appropriate on each data set in turn
- We then combine the results of those multiple analyses using fairly simple rules
- In a way it is like running multiple replications of an experiment and then combining the results across the multiple analyses
- But in the case of MI, the replications are repeated simulations of data sets based upon parameter estimates from the original study

Multiple Imputation with R package MICE: R Code

```
> fitness<-read.table("C://documents and settings//
+         krishnan.t//desktop//fitness2.txt",header=TRUE)
> require(mice)
> require(lattice)
> imp<-mice(fitness,m=5,maxit=2)
> mat<-complete(imp)
> mat
> bwplot(imp)
```

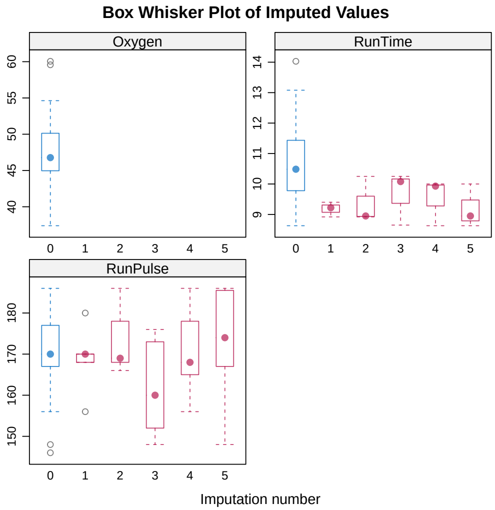
Multiple Imputation with R Package MICE: R Output

```
iter imp variable
  1    1   RunTime   RunPulse
  1    2   RunTime   RunPulse
  1    3   RunTime   RunPulse
  1    4   RunTime   RunPulse
  1    5   RunTime   RunPulse
  2    1   RunTime   RunPulse
  2    2   RunTime   RunPulse
  2    3   RunTime   RunPulse
  2    4   RunTime   RunPulse
  2    5   RunTime   RunPulse
```

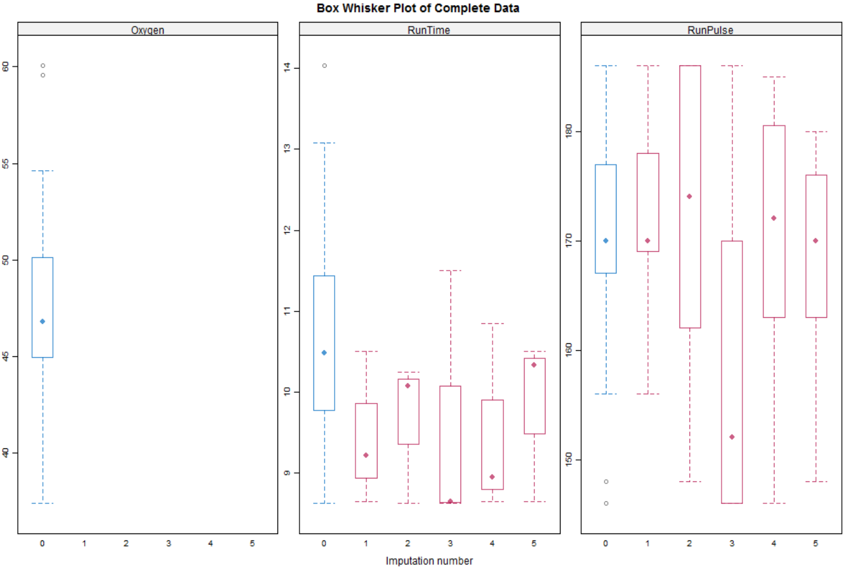
Multiple Imputation with R Package MICE: R Output

```
   Oxygen RunTime RunPulse
1  44.609   11.37      178
2  45.313   10.07      185
3  54.297    8.65      156
4  59.571    8.92      168
5  49.874    9.22      170
6  44.811   11.63      176
7  45.681   11.95      176
8  49.091   10.85      180
9  39.442   13.08      174
10 60.055    8.63      170
11 50.541    9.40      170
12 37.388   14.03      186
13 44.754   11.12      176
14 47.273    9.22      168
15 51.855   10.33      166
16 49.156    8.95      180
17 40.836   10.95      168
18 46.672   10.00      170
19 46.774   10.25      170
20 50.388   10.08      168
21 39.407   12.63      174
22 46.080   11.17      156
23 45.441    9.63      164
24 54.625    8.92      146
25 45.118   11.08      156
26 39.203   12.88      168
27 45.790   10.47      186
28 50.545    9.93      148
29 48.673    9.40      186
30 47.920   11.50      170
```

Multiple Imputation with R Package MICE: R Output



Box Whisker Plot of Completed Fitness Data



Little's Test for MCAR

- Segregate various missing value patterns
- Maximum likelihood estimates of means are computed
- For each pattern, compare observed variable mean vector with MLE of it, weighted by covariances
- Compute overall weighted squared deviation
- Use that as a chi-squared statistic
- Rationale: If each pattern produces a different mean, MCAR is unlikely
- The degrees of freedom: the number of variables for each pattern − the total number of variables
- Small p-value would imply NOT MCAR

Little's Test for MCAR: R Code

```
> require(BaylorEdPsych)
> LittleMCAR(fitness)
```

Little's Test for MCAR: R Output

```
this could take a while[1] 3.968521

this could take a while[1] 3

this could take a while[1] 0.2648834

this could take a while[1] 3

this could take a while                Oxygen    RunTime  RunP
Number Missing        0 3.00000000 8.0000000
Percent Missing       0 0.09677419 0.2580645
```

p-value being 0.26, the hypothesis of MCAR is not rejected

Little's Test for MCAR: R Output: Complete Data

```
this could take a while$DataSet1
   Oxygen RunTime RunPulse
1  44.609   11.37      178
2  45.313   10.07      185
3  54.297    8.65      156
6  44.811   11.63      176
7  45.681   11.95      176
9  39.442   13.08      174
10 60.055    8.63      170
12 37.388   14.03      186
13 44.754   11.12      176
15 51.855   10.33      166
16 49.156    8.95      180
17 40.836   10.95      168
20 50.388   10.08      168
21 39.407   12.63      174
22 46.080   11.17      156
23 45.441    9.63      164
24 54.625    8.92      146
26 39.203   12.88      168
27 45.790   10.47      186
28 50.545    9.93      148
29 48.673    9.40      186
30 47.920   11.50      170
31 47.467   10.50      170

$DataSet2
   Oxygen RunTime RunPulse
5  49.874    9.22       NA
8  49.091   10.85       NA
18 46.672   10.00       NA
19 46.774   10.25       NA
25 45.118   11.08       NA
```