Regression Analysis

Final Project

By: Ganga Acharya, Sara Biesiadny, and Matthew Tichenor

This paper is organized into eight different sections. First the mechanisms of missing data, i.e. the types of missingness, and patterns of missing data are introduced, followed by a brief introduction to simple methods for handling missing data. A simulation is done to compare the methods for a Missing Completely at Random dataset. The Expectation Maximization algorithm is then discussed followed by an introduction to Multiple Imputation and imputation algorithms for Missing at Random data with a Monotone Missing Pattern. A tennis dataset with a Monotone Missing Pattern was found to demonstrate the Expectation Maximization algorithm and the Multiple Imputation algorithms previously discussed. Statistics unique to Multiple Imputation are also discussed, in the results and discussion section for this application. Finally the programming scripts used in this final project are revealed. Excel VBA, R, and SAS were all used to obtain results.

Types of Missingness

For any method of handling missing data, it's important to identify or make assumptions about the way the data is missing. In general, there are three ways that a dataset can have missing values. The values may be missing completely at random, missing at random, or missing not at random.

Data is missing completely at random (MCAR) if the missingness of a variable does not depend upon any variables, observed or unobserved, in the dataset. This definition can be made more precise with an example. Suppose that a dataset has only one missing variable Y, which has a total of n missing and observed values. Define R to be an indicator variable for the missingness of Y, where $R_i = 1$ if the observation Y_i is missing and $R = 0$ otherwise for $i = 1, \dots, n$. Let X indicate the completely observed variables from the dataset. The data is missing completely at random if and only if $P(R = 1|X, Y) = P(R = 1|X) = P(R = 1)$.

Continuing this example, whenever $P(R = 1|X, Y) = P(R = 1|X)$, the data is missing at random (MAR). In other words, data is missing at random whenever the missingness of a variable does not depend on the variable with missing values. With this definition, missing completely at random is just a special case of missing at random.

If we allow the missing data to depend on the variable with missing data, then the data is called missing not random (MNAR). Continuing the example from before, the data is missing not at random if we cannot simplify $P(R = 1|X, Y)$. While there are some special situations MNAR data is desirable, many methods of imputation rely on a MAR assumption. It is unfortunate that, currently, there is no way to verify the MAR assumption for a dataset, but this assumption is more believable as the size of the dataset and number of variables increases (Wikipedia, "Missing Data").

Complementing the types of missingness are the various types of missing patterns that may be encountered. The missing data patterns should not be confused with the mechanism for missing data e.g. MCAR. Missing data patterns are not probabilistic statements, just simple ways to describe a dataset.

If there is only one variable missing in the dataset, then the data has a Univariate Missing Pattern. When only one group of one or more variables are always simultaneously missing, then the dataset has a Unit Non-Response Pattern. In other words, rather than having a single missing variable, a group of variables, i.e. a unit, is acting as the one missing variable. *Figure 1* shows pictures to illustrate the idea.

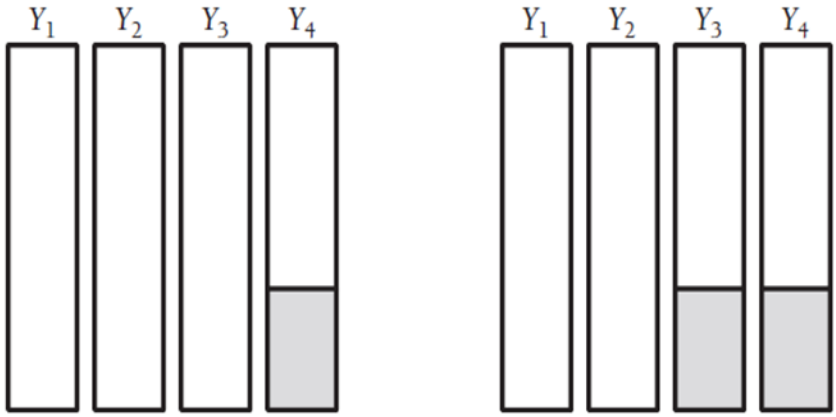


Figure 1: Univariate & Unit Non-Response Patterns

A Monotone Missing Pattern and Arbitrary Missing Pattern are shown in *Figure 2*. A Monotone Missing Pattern occurs whenever subsequent variables are missing, following a missing variable. The Monotone Missing Pattern is special, because it allows simpler imputation methods than an Arbitrary Pattern. As the name implies, an Arbitrary Missing Pattern occurs when the data appears missing in a random fashion. Like the image in *Figure 2*, you can think of an Arbitrary Missing Pattern as resembling Swiss cheese.

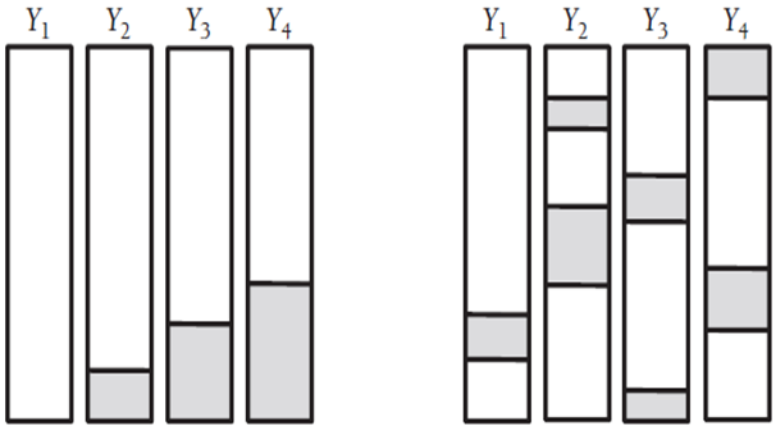


Figure 2: Monotone & Arbitrary Patterns

Two other missing data patterns that can occur are the Planned Pattern and Latent Variable Pattern. These patterns are described by *Figure 3*. Both of these patterns occur intentionally, so they won't be emphasized in this report.

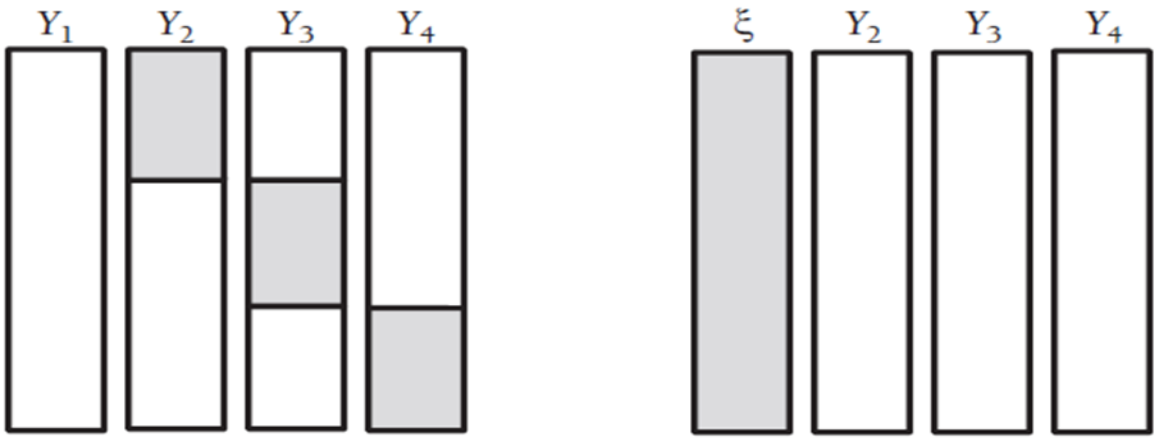


Figure 3: Planned & Latent Variable Pattern

Simple Ways for Handling Missing Values

There are a variety of simple ways for handling missing data. While these methods are not recommended, they have been used in the past.

One of the first things that comes to mind when viewing missing data, is to ignore the problem. List-wise deletion essentially does this. In list-wise deletion, observations which contain missing values are removed from the dataset so analysis can continue. One problem with List-wise deletion, is that it reduces the sample size and therefore the power of a statistical test. In addition, parameter estimates can be biased when the data is not MCAR (Enders, 2010).

Another alternative is to fill in the missing values with estimates. Filling in missing values is called imputation, and will be referred to as imputation throughout the rest of this paper. The simplest methods of imputation are called single imputation, because the missing values are imputed only once. Mean Imputation, Regression Mean Imputation and Stochastic Regression Mean Imputation are all single imputation methods.

Mean Imputation imputes the missing values with the mean of the observed variables. In regression where a design matrix is constructed, the imputed values for a variable would be the column means. While this method is simple, it adversely affects measures of association between covariates. For example, the sample covariance is given by the formula $\hat{\sigma}_{xy} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n-1}$. Replacing a missing value with its mean results in a value of 0 in the summation. Repeatedly replacing missing values with their corresponding means will drastically reduce the sample covariance. In addition biases are present under the assumption of any missing data mechanism (Enders, 2010).

Regression Mean and Stochastic Regression Mean Imputation impute the missing values by regressing the missing variable with completely observed covariates. Regression Mean Imputation imputes the missing variable with the regression mean. For example, if the variable Y is missing, and X is a matrix of completely observed random variables, then regression coefficients, $\hat{\beta}$, are estimated from the model $Y = X\beta + \epsilon$, i.e. the multiple least squares model with standard assumptions, using the observed values of Y. Using the estimates, the missing values for Y are imputed in using the equation $Y = X\hat{\beta}$, i.e. the regression mean. Stochastic Regression Mean Imputation would follow the same procedure except that

the missing values would be imputed with $Y = X\hat{\beta} + z$, where z is a vector of independent, random draws from a standard normal distribution. The problem with Regression Mean Imputation, which Stochastic Regression Mean Imputation addresses, is that the regression model assumes a relationship between covariates, and imputes the missing values with values that are highly correlated with the completely observed covariates. Stochastic Regression Mean Imputation reduces the artificially high correlation by introducing a random error term.

While literature on the subject has plenty of examples showing the problems with simple imputation methods, experience is the best teacher. A simulation was made to calculate the bias, variance, and mean square error for regression estimates after using these four simple methods.

For the simulation, the response variable y is the a linear combination of independent covariates x_1, x_2, and x_3, plus an error term, where $X_1 \sim N(6,1)$, $X_2 \sim N(0,1)$, and $X_3 \sim N(-10,1)$; $y = 1 + 1x_1 - 1x_2 + 1x_3 + \epsilon$, and $\epsilon \sim N(0,1)$. 250 observations were generated using R functions to create the dataset.

Next, X_3's observations were deleted from the complete dataset, in a random way for 10,000 iterations to create 10,000 datasets with missing values. The probability of missingness for X_3, was a Bernoulli random variable with probability of missingness, $p = 0.3$. Because the probability of missingness does not depend on any variable from the dataset, the mechanism for missing data is MCAR.

For each of the 10,000 MCAR datasets, regression estimates for the multiple linear regression model $Y = X\beta + \epsilon$, were estimated, and collected into a matrix of $\hat{\beta}$'s for each method. Altogether there were four matrices full of regression coefficients for the four simple methods: List-wise Deletion, Mean Imputation, Regression Mean Imputation, and Stochastic Mean Regression Imputation.

The advantage of a simulation is that the true values are known. For this simulation, $\beta = (1,1,-1,1)^T$. With the true values known, the bias, variance, and mean square error for the regression estimates can be calculated. The bias for a regression coefficient is $bias = (\overline{\beta_i} - \beta_i)$, for $i = 1, \dots, p$, where $\overline{\beta_i} = \frac{1}{n}\sum\widehat{\beta_{ij}}$. The variance for a regression coefficient is the sample variance for the coefficient, $var = \frac{1}{n-1}\sum\left(\beta_{ij} - \bar{\beta}\right)^2$, for $i = 1, \dots, p$. Likewise, the standard error is the square root of the sample variance. Finally, the estimate for the mean square error is $mse = bias^2 + var$. The table below lists these three statistics for each regression coefficient for each method.

		Method			
Statistics		**List-Wise Deletion**	**Mean Imputation**	**Regression Mean Imputation**	**Stochastic Regression Mean Imputation**
bias	intercept	-2.2806992	-2.4247228	-12.49792946	-12.49828979
	X1	0.01337073	0.03805876	0.08809639	0.08810158
	X2	-0.0332110	-0.0273975	-0.01093928	-0.01095588
	X3	-0.0230649	-0.0227010	-0.98631352	-0.98637205
standard error	intercept	0.48718556	0.48521607	0.110522073	0.110314516
	X1	0.04398384	0.02809106	0.009371896	0.009353041
	X2	0.03757761	0.02534665	0.008419968	0.008411341
	X3	0.04215475	0.04251008	0.012602017	0.012574046

MSE	interc ept	5.438938765	6.114715059	1.56E+02	1.56E+02
	X1	0.002113354	0.002237577	7.85E-03	7.85E-03
	X2	0.002515046	0.001393075	1.91E-04	1.91E-04
	X3	0.002309012	0.002322441	9.73E-01	9.73E-01

From the table some interesting results appear for this MCAR simulation. List-wise deletion and Mean Imputation produce less biased estimates for all regression coefficients except for the coefficient corresponding to X_2. On the other hand, Regression Mean Imputation and Stochastic Regression Mean Imputation estimates have a smaller standard error than List-wise deletion and Mean Imputation. Combining these two statistics is the estimated mean square error. For the Regression Mean and Stochastic Regression Mean Imputation, the mse is smaller for the X_2 regression coefficient, but Mean Imputation and List-wise deletion have a smaller mse for every other regression coefficient. Probably the most condemning piece of information is the mse for the intercept. The mse for the Regression Mean Imputation and Stochastic Regression Mean Imputation is approximately 156! This is far too large to warrant use in the MCAR situation. Since List-wise deletion has the smallest mse for the majority of the regression coefficients, this method should be used whenever the MCAR assumption is true and the statistical power of a test is not too adversely affected.

While this simulation has compared single imputation methods under MCAR conditions, imputation methods are also needed for the MAR mechanism. While it is true that if data is MCAR then it is MAR, the converse is not true. More sophisticated methods have been developed for imputation under the MAR assumption and certain missing data patterns.

E-M Algorithm

The Expectation Maximization (E-M) algorithm is a two-step iterative procedure, made up of the E-step and the M-step. The E-step uses the initial estimates of the mean vector and covariance matrix to build a set of regression equations that predicts the missing data from the observed data. The M-step applies the complete-data formula to the filled in data to generate the updated estimates of the mean vector and covariance matrix. The algorithm is adjusted so that it carries out the updated mean vector and covariance matrix in the next step to build a new set of regression equations to predict the missing values. The subsequent M-step estimates the mean and covariance matrix again for the updated data set. This process ends when there is no longer a change in values of mean and covariance in subsequent steps and we say that the algorithm converged to the Maximum Likelihood Estimate (MLE). Since the E-M algorithm is accompanied by Maximum Likelihood Estimation (MLE), MLE yields unbiased estimates of parameters for the MAR data; the E-M algorithm yields unbiased parameters for our MAR data. Thus we decided to use the E-M algorithm to fix the problem of missingness in our data set.

In the case of the bivariate distribution with missing values in the Y variable, we use the following formulas on the complete data set (filled in using the E-M algorithm) to find the maximum likelihood of the mean and covariance.

$$\hat{\mu}_y = \frac{1}{N}\sum Y$$

$$\hat{\sigma}_y^2 = \frac{1}{N}(\sum Y^2 - \frac{1}{N}(\sum Y)^2)$$

$$\hat{\sigma}_{x,y}^2 = \frac{1}{N}(\sum XY - \frac{\sum X \sum Y}{N})$$

We use the following equations for regression equations.

$$\hat{\beta}_1 = \frac{\hat{\sigma}_{x,y}}{\hat{\sigma}_x^2}$$

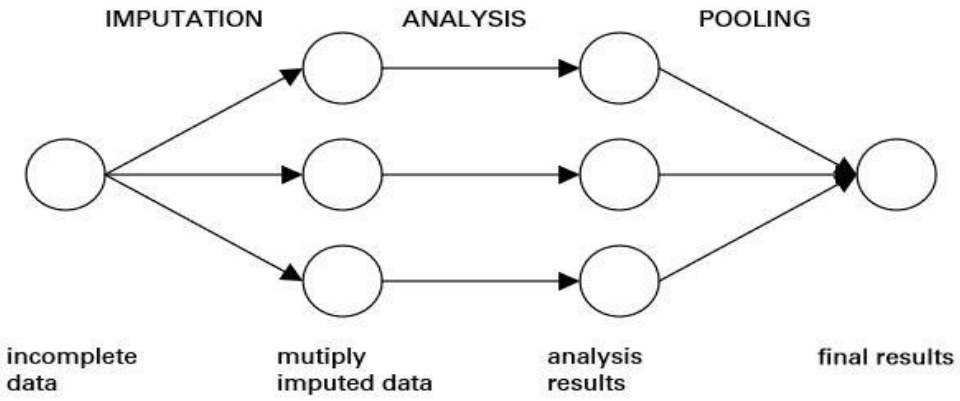
$$\hat{\beta}_0 = \hat{\mu}_y - \hat{\beta}_1 \hat{\mu}_x$$

$$\hat{\sigma}_{Y|X}^2 = \hat{\sigma}_x{}^2 - \hat{\beta}_1^2 \hat{\sigma}_x^2$$

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$$

Where $\hat{\beta}_1$ and $\hat{\beta}_0$ are the slope and intercept coefficients, respectively, and $\hat{\sigma}_{Y|X}^2$ is the residual variance from the regression of Y on X. For the multivariate data we extend the above formula in the same manner.

Multiple Imputation



Another technique for analyzing incomplete datasets is *multiple imputation*. In using the technique of multiple imputation, a dataset with missingness is imputed not once, but *m* times. In the figure above, we have $m = 3$. As is shown in the figure, multiple imputation consists of three major steps: *imputation*, *analysis*, and *pooling*. In the imputation step, all missing entries in the dataset are filled in (imputed) *m* times, drawing from some distribution (which can be different for each missing entry) to complete the dataset. After the dataset's missing values are imputed, analysis is then performed on each one of the *m* (now complete) datasets. By "analysis" here, in our case we mean that an estimate of each regression coefficient is computed from each one of the *m* datasets. The pooling step is the problematic step in the process. We want to find some way to pool our *m* estimates together to create an overall estimate with as little error as possible. There are many ways to perform this pooling, but one very common way is through *Rubin's Rules* (1987).

To use Rubin's rules, we first need to define some notation. Let us let $\hat{Q}_j$ denote an estimate of a scalar quantity of interest (e.g. a regression coefficient) from the j^{th} imputation and let U_j denote its corresponding standard error. It is trivial here that $j = 1, \dots, m$ (where we have *m* imputations). Once we have computed each one of our *m* values for $\hat{Q}_j$, we can calculate an overall estimate of our scalar quantity of interest by simply taking the average of our *m* individual estimates. If we denote this overall estimate by $\bar{Q}$, we have $\bar{Q} = \frac{1}{m}\sum_{j=1}^{m} \hat{Q}_j$. To calculate the variance of our estimates, we first have to recognize that there are two types of variance present: within-imputation variance and between-imputation variance. We can easily compute the within-imputation variance by again simply taking the average, this time of our individual standard errors. If we denote the within-imputation variance by $\bar{U}$, we have $\bar{U} = \frac{1}{m}\sum_{j=1}^{m} U_j$. The between-imputation variance is a bit more complicated, but it turns out to just be the sample variance of the estimates. If we denote the between-imputation variance by *B*, we have $B = \frac{1}{m-1}\sum_{j=1}^{m}\left(\hat{Q}_j - \bar{Q}\right)^2$. Once we have defined these two quantities for within-imputation variance and between-imputation variance, we can define the total variance, *T*, by $T = \bar{U} + \left(1 + \frac{1}{m}\right)B$. Here, *T* is more or less a weighted sum of the two types of variance present. If we denote the total variance by *T*, we can then say that the overall standard error is $\sqrt{T}$.

Once the overall standard error, $\sqrt{T}$, is defined, confidence intervals and hypothesis tests for the true value of the quantity can be computed. Confidence intervals for *Q* can be constructed by $\bar{Q} \pm t_\Delta\sqrt{T}$, where t_Δ is a quantile of the Student's t-distribution with $\Delta = (m-1)\left(1 + \frac{m\bar{U}}{(m+1)B}\right)^2$ degrees of freedom. In addition, hypothesis tests for testing $H_0: Q = 0$ can be performed by computing the test statistic $t = \frac{\bar{Q}}{\sqrt{T}}$ and comparing it to the same quantile, t_Δ. If $t > t_\Delta$, then the null hypothesis is rejected.

Imputation Algorithms

Under the MAR assumption, a dataset with a monotone missing pattern may have multiple imputed datasets generated using the Monotone Regression Method, Monotone Regression Predictive Mean Matching Method or the Propensity Score Method (SAS/STAT(R) 9.3 User's Guide, 2009). While the Propensity Score Method is a non-parametric imputation method, the Monotone Regression Method and Monotone Regression Predictive Mean Matching Method both assume a multivariate normal distribution for the independent variables with missing data. As the name suggests, both the Monotone Regression Method and Monotone Regression Predictive Mean Matching Method involve a regression equation.

For the Monotone Regression Method algorithm begin by fitting a regression equation for the variable with the least number of missing values. In *figure 2*, this would be the variable Y_2. For this variable, we obtain regression coefficients using the observed values of the missing variable and completely observed covariates, i.e. find the values $\hat{\beta}$ for the equation $Y = X\beta + \epsilon$, where Y is a vector of the observed values and X is a matrix containing the corresponding observations for the completely observed covariates.

Using the coefficient estimates $\hat{\beta}$ the next step is to draw new values from the posterior predictive distribution, $\beta|X, Y$. Draw σ_{*j}^2 from the distribution $\frac{\hat{\sigma}_j^2(n_j-k-1)}{g}$, where n_j is the number of non-missing

observations for Y_j, $g \sim X^2_{n_j-k-1}$ and $\hat{\sigma}_j^2$ is the squared residual standard error from the previous regression. Next, draw $\widehat{\beta_*}$ from $\hat{\beta} + \sigma_{*j} V'_{hj} Z$, where V'_{hj} is a vector of independent random normal variates from the choleski decomposition $V'_{hj} V_{hj} = X'X$, Z and $\hat{\beta}$ is the vector of parameter estimates from the previous regression.

The final step is to use the new regression coefficients $\widehat{\beta_*}$ to impute the missing values. That is, for the i^{th} observation for which Y is missing, impute the missing values by using the stochastic regression equation $y_{*i} = \beta_{*0} + \beta_{*1} x_1 + \cdots + \beta_{*k} x_k + z_i \sigma_{*j}$, where z_i is a draw from an independent standard normal variate (SAS/STAT(R) 9.3 User's Guide, 2009).

Now that one of missing variables has been imputed, you can repeat the process for the next variable with missing values. Referring to *figure 2*, once we've imputed values for Y_2, we can consider Y_2 to be one of the completely observed variables, and repeat the process for Y_3. This process is repeated for each missing variable, so that all monotonically missing variables have been imputed; and once all values have been imputed, the entire process is repeated m times.

The Monotone Regression Predictive Mean Matching Method is very similar to the previous imputation method. In fact, the algorithm is identical, with the exception of a few additional steps. As before, the first step is to fit a regression equation to the observed values of the variable with missing values Y, i.e. the response variable is the observed values of Y and the explanatory variables are the corresponding, completely observed, values. Once the regression coefficients have been estimated, draw new coefficients from the posterior predictive distribution, $\beta|X, Y$. For each missing value, find a predicted value with the regression equation $y_{*i} = \beta_{*0} + \beta_{*1} x_1 + \cdots + \beta_{*k} x_k$, where $\widehat{\beta_*}$ are the draws from $\beta|X, Y$. For each predicted value, create a list of k observed values of Y, for which $\beta_{*0} + \beta_{*1} x_1 + \cdots + \beta_{*k} x_k$ is closest to the predicted value, then from the list randomly choose one of the k values to impute the missing value (SAS/STAT(R) 9.3 User's Guide, 2009).

Just as before, once all missing values for a variable have been imputed, the process would be repeated for the next monotone missing variable, until all missing values have been imputed. This entire process is repeated m times to form m different multiply imputed datasets.

Application

To apply these techniques to a dataset, we chose to analyze a dataset corresponding to tennis data from the four major tennis tournaments in the year 2013. The data we chose to analyze has missing values that will be talked about below. Scoring in professional tennis is rather complicated, but there are some introductory guidelines and terms that need to be introduced before the variables will make sense. A tennis *match* (the entire contest between two opponents) is composed of a number of *sets*. In men's major tennis tournaments, the first player or team to win *three* sets wins the match. In women's major tennis tournaments, the first player or team to win *two* sets wins the match. Each set is made up of a number of *games*. The first player or team to win six games (must win by two; cannot win 6-5) wins the set. The firsr player or team to achieve four points (again, must win by two) wins the game. Gameplay in tennis begins with a *serve*, and when the servers opponent is unable to successfully return valid serve it is called an *ace* for the server. When a shot is hit cleanly past an opponent (who does not touch it) or when the ball bounces twice on on side of the court, it is called a *winner*. An error in a service or return shot that cannot be attributed to any factor other than poor judgement and execution by the player is

called an *unforced error*. Finally, a *break point* is a point which, if won by the person or team receiving the serve, sould result in a break of service (the person or team that was receiving will now start serving). Now that we have defined these terms and guidelines, the varaibles used in our data set are provided below:

- FNL.1: Final Number of Games Won by Player 1 (This is our response variable)
- FSW.1: First Serve Won by Player 1
- SSW.1: Second Serve Won by Player 1
- ACE.1: Aces Won by Player 1
- WNR.1: Winners Earned by Player 1
- UFE.1: Unforced Errors Committed by Player 1
- BPC.1: Break Points Created by Player 1
- BPW.1: Break Points Won by Player 1
- TPW.1: Total Points Won by Player 1
- ST*j*.1: Set *j* Result for Player 1 ($j = 1, \dots, 5$)

Using these variables, there is a mononte missing pattern for the data. *Figure 4* presents a table which illustrates the pattern. In *Figure 4*, the 2nd through 15th column names represent variables and row names represent various groups of missing data. We see in Group 1, a pattern $X, X, \dots, X$, where an X indicates that the variable is completely observed. This means Group 1 is the subset of observations which were completely observed. The frequency and relative frequency for Group 1 was 21 and 11.29%, with the interpretation that only 21 observations, or 11.29% of the data, from the tennis dataset were completely observed. For Group 2, there is a sequence of X's followed by a period, . , where the period indicates a missing value for a variables. Group 2 is the subset of observations where all variables, except for ST51 are missing. Examining all groups, the pattern here is that, if a variable is missing, then all subsequent variables are missing as well, so our dataset with the following variables chosen, has a monotone missing pattern.

Group	FNL1	FSW1	SSW1	ACE1	WNR1	UFE1	BPC1	BPW1	TPW1	ST11	ST21	ST31	ST41	ST51	Freq	Percent
1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	21	11.29
2	X	X	X	X	X	X	X	X	X	X	X	X	X	.	38	20.43
3	X	X	X	X	X	X	X	X	X	X	X	X	.	.	79	42.47
4	X	X	X	X	X	X	X	X	X	X	X	.	.	.	47	25.27
5	X	X	X	X	X	X	X	X	X	X	.	.	.	.	1	0.54

Figure 4:The Pattern of Missing Values

Because the dataset has a monotone missing pattern, the Monotone Regression Method and Monotone Regression Predictive Mean Matching Methods was used to multiply impute the dataset and obtain confidence intervals regression coefficients. In addition, the regression coefficients from the E-M algorithm were also calculated and are discussed in the next section.

Results and Discussion

Using the Monotone Regression Method first to impute the dataset $m = 5$ times, the regression coefficients were estimated for the model $FNL1 = \mathrm{B}_0 + \beta_1 FSW1 + \mathrm{B}_2 SSW1 + \beta_3 ACE1 + \beta_4 WRN1 + \beta_5 UFE1 + \beta_6 BPC1 + \beta_7 BPW1 + \beta_8 TPW1 + \beta_9 ST11 + \beta_{10} ST21 + \beta_{11} ST31 + \beta_{12} ST41 + \beta_{13} ST51 + \epsilon$. The m sets of estimates were combined using Rubin's Rules.

Figure 5 shows SAS output which displays information about the variance as well as a few other statistics. We see the between, within and total variance which are calculated using Rubin's Rules, for all 13 regression coefficients. We also see the degrees of freedom for each regression coefficient. There are three additional statistics displayed in *Figure 5*: the Relative Increase in Variance, the Fraction of Missing Information and the Relative Efficiency. These statistics are important enough for their own explanation and discussion.

Define the Relative Increase in Variance due to Non-Response to be $\mathrm{r} = \frac{(1+m^{-1})B}{\bar{W}}$, where B is the between imputation variance and $\bar{W}$ is the within imputation variance. The r yields the proportional increase in the sampling variance that is due to missing data (Enders 2010, pp. 225). Define the Fraction of Missing Information to be $\hat{\lambda} = \frac{r+2/(v_m+3)}{r+1}$, where v_m is the degrees of freedom for the *t*-test statistic. The $\hat{\lambda}$ gives the proportion of the total sampling variance that is due to missing data (Enders 2010, pp.

226). Define the Relative Efficiency to be $RE = (1 + \frac{\lambda}{m})^{-1}$. RE quantifies the magnitude of a multiple imputation standard error relative the hypothetical minimum, achieved by an infinite number of imputations! For example, if $\lambda = 0.2$ and $m = 5$, then $RE = 96\%$, meaning the hypothetical minimum standard error is 96% as large as the standard error given 5 imputations (Enders 2010, pp. 213).

Pooling the Parameter Estimates

The MIANALYZE Procedure

Model Information	
Data Set	WORK.OUTREG1
Number of Imputations	5

Variance Information

Parameter	Variance			DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
	Between	Within	Total				
Intercept	0.018028	0.122733	0.144367	118.84	0.176268	0.159241	0.969135
FSW1	0.000011042	0.000164	0.000177	252.03	0.080723	0.077264	0.984782
SSW1	0.000020940	0.000208	0.000233	179.37	0.120852	0.112962	0.977907
ACE1	0.000009977	0.000129	0.000141	227.16	0.092733	0.088141	0.982677
WNR1	0.000009494	0.000043934	0.000055327	73.539	0.259307	0.222228	0.957446
UFE1	0.000010969	0.000027923	0.000041086	34.291	0.471384	0.352752	0.934099
BPC1	0.001279	0.002878	0.004412	29.507	0.533246	0.383953	0.928686
BPW1	0.000065365	0.000346	0.000424	87.265	0.226709	0.198385	0.961837
TPW1	0.000002836	0.000077728	0.000081131	341.87	0.043786	0.042791	0.991514
ST11	0.001067	0.002478	0.003758	30.664	0.516514	0.375779	0.930098
ST21	0.000386	0.001321	0.001785	49.821	0.350730	0.283416	0.946357
ST31	0.000373	0.001241	0.001690	48.014	0.360971	0.289779	0.945219
ST41	0.000319	0.000796	0.001180	33.426	0.481269	0.357922	0.933198
ST51	0.000595	0.000630	0.001343	13.222	1.133252	0.585823	0.895123

Figure 5: Monotone Regression Method Variance Info

Returning to *Figure 5*, we can now accurately interpret the last three statistics. We see that the regression coefficient for the variable $ST51$has the largest Fraction of Missing Information, which is not surprising, since this was the most missing variable. We also see the Relative Efficiency for this parameter estimate is 89.5%, meaning the hypothetical minimum variance is 89.5% as large as variance with 5 imputations. For the other coefficients the Relative Efficiency is larger than 92%.

Figure 6 is another table output from SAS. *Figure 6* displays the estimates, standard error, 95% confidence intervals and *p*-values for the regression coefficients. Examining the *p*-values, regression coefficients corresponding to the intercept and variables $WRN1, BPC1$ and $ST21$ are significant at the 5% significance level. It is also important to note that these results are valid under the MAR and multivariate normality assumption, and may not be robust to violations of these assumptions.

Parameter Estimates										
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	-1.303226	0.379957	-2.05559	-0.55086	118.84	-1.440311	-1.092432	0	-3.43	0.0008
FSW1	0.021284	0.013319	-0.00495	0.04751	252.03	0.017854	0.024709	0	1.60	0.1113
SSW1	0.017222	0.015266	-0.01290	0.04735	179.37	0.012746	0.022524	0	1.13	0.2608
ACE1	0.012119	0.011878	-0.01128	0.03552	227.16	0.007384	0.015192	0	1.02	0.3086
WNR1	0.014829	0.007438	0.00001	0.02965	73.539	0.010152	0.017693	0	1.99	0.0499
UFE1	-0.012261	0.006410	-0.02528	0.00076	34.291	-0.015615	-0.007565	0	-1.91	0.0641
BPC1	0.242243	0.066423	0.10649	0.37799	29.507	0.196869	0.281280	0	3.65	0.0010
BPW1	0.011977	0.020602	-0.02897	0.05292	87.265	0.001051	0.018941	0	0.58	0.5625
TPW1	-0.012901	0.009007	-0.03062	0.00482	341.87	-0.015803	-0.011467	0	-1.43	0.1530
ST11	0.085738	0.061302	-0.03934	0.21082	30.664	0.046773	0.117867	0	1.40	0.1720
ST21	0.155508	0.042246	0.07065	0.24037	49.821	0.133090	0.184090	0	3.68	0.0006
ST31	0.102877	0.041104	0.02023	0.18552	48.014	0.078284	0.124592	0	2.50	0.0158
ST41	0.049435	0.034347	-0.02041	0.11928	33.426	0.024179	0.066722	0	1.44	0.1594
ST51	-0.001077	0.036651	-0.08012	0.07797	13.222	-0.029075	0.029039	0	-0.03	0.9770

Figure 6: Monotone Regression Parameter Estimates

In addition to the Monotone Regression Method, analysis was carried out using the Monotone Regression Predictive Mean Matching Method. Using this method 5 multiply impute datasets were generated, then the regression coefficients were estimated for the model $FNL1 = \mathrm{B}_0 + \beta_1 FSW1 + \mathrm{B}_2 SSW1 + \beta_3 ACE1 + \beta_4 WRN1 + \beta_5 UFE1 + \beta_6 BPC1 + \beta_7 BPW1 + \beta_8 TPW1 + \beta_9 ST11 + \beta_{10} ST21 + \beta_{11} ST31 + \beta_{12} ST41 + \beta_{13} ST51 + \epsilon$. The m sets of estimates were combined using Rubin's Rules.

The variance information for the parameter estimates is given in *Figure 7*. We see that this time the Relative Efficiencies for the parameters estimates are not quite as good. The regression coefficients corresponding to the variables $ST41$ and $ST51$ have approximately 88% Relative Efficiency. This suggests that more than 5 imputations would lower the total variance for the parameter estimates. If we compare the total variance for $ST41$ and $ST51$ in *Figure 7* with those in *Figure 5*, we see that the Monotone Regression Predictive Mean Matching Method achieves smaller total variances than the Monotone Regression Method. For this reason, and to make a better comparison between methods, 5 imputations is sufficient.

Pooling the Parameter Estimates

The MIANALYZE Procedure

Model Information	
Data Set	WORK.OUTREG2
Number of Imputations	5

Variance Information

Parameter	Variance			DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
	Between	Within	Total				
Intercept	0.019324	0.025713	0.048902	16.463	0.901841	0.524779	0.905014
FSW1	0.000012366	0.000063292	0.000078131	83.632	0.234456	0.204152	0.960771
SSW1	0.000002816	0.000069051	0.000072430	328.8	0.048931	0.047684	0.990553
ACE1	0.000000763	0.000045430	0.000046346	395.81	0.020156	0.019949	0.996026
WNR1	0.000002375	0.000011655	0.000014505	79.279	0.244517	0.211548	0.959408
UFE1	0.000003633	0.000008409	0.000012768	30.53	0.518387	0.376703	0.929938
BPC1	0.000397	0.000633	0.001109	19.9	0.752286	0.475522	0.913155
BPW1	0.000010044	0.000123	0.000135	217.37	0.097864	0.092738	0.981790
TPW1	0.000002729	0.000030232	0.000033507	198.95	0.108314	0.102007	0.980006
ST11	0.000120	0.000455	0.000598	56.997	0.315956	0.261091	0.950373
ST21	0.000156	0.000435	0.000623	38.302	0.431298	0.331031	0.937905
ST31	0.000084220	0.000340	0.000441	61.711	0.297107	0.248512	0.952651
ST41	0.000296	0.000248	0.000603	10.816	1.431819	0.645355	0.885684
ST51	0.000362	0.000262	0.000697	9.6377	1.661360	0.680907	0.880141

Figure 7: Predictive Mean Matching Variance Info

Figure 8 displays more SAS output. The parameter estimates, standard errors, 95% confidence intervals and *p*-values for the regression coefficients, using the Monotone Regression Predictive Mean Matching Method are shown. Examining the *p*-values for the individual regression coefficients we see that the intercept and coefficients corresponding to the variables $WRN1, UFE1, BPC1, ST11, ST21$ and $ST31$ are all significant at the 5% significance level. This contrasts with the results from the Monotone Regression Method where coefficients corresponding to the $ST11$ and $ST31$ variables were not significant. This difference can be attributed to the smaller standard errors achieved using the Monotone Regression Predictive Mean Matching Method.

Parameter Estimates										
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	-1.503348	0.221137	-1.97107	-1.03563	16.463	-1.695460	-1.389775	0	-6.80	<.0001
FSW1	0.013471	0.008839	-0.00411	0.03105	83.632	0.007981	0.017582	0	1.52	0.1313
SSW1	0.001365	0.008511	-0.01538	0.01811	328.8	-0.000441	0.003231	0	0.16	0.8727
ACE1	0.012124	0.006808	-0.00126	0.02551	395.81	0.010623	0.012808	0	1.78	0.0757
WNR1	0.010417	0.003809	0.00284	0.01800	79.279	0.008097	0.012236	0	2.74	0.0077
UFE1	-0.008807	0.003573	-0.01610	-0.00151	30.53	-0.010846	-0.005899	0	-2.46	0.0195
BPC1	0.227598	0.033308	0.15810	0.29710	19.9	0.198697	0.249322	0	6.83	<.0001
BPW1	0.011356	0.011628	-0.01156	0.03427	217.37	0.007236	0.016064	0	0.98	0.3298
TPW1	-0.005435	0.005789	-0.01685	0.00598	198.95	-0.007392	-0.003883	0	-0.94	0.3489
ST11	0.103653	0.024464	0.05467	0.15264	56.997	0.089714	0.118748	0	4.24	<.0001
ST21	0.144979	0.024951	0.09448	0.19548	38.302	0.134943	0.164897	0	5.81	<.0001
ST31	0.114460	0.021005	0.07247	0.15645	61.711	0.104204	0.127630	0	5.45	<.0001
ST41	0.040150	0.024548	-0.01399	0.09429	10.816	0.028649	0.070113	0	1.64	0.1307
ST51	0.031813	0.026392	-0.02729	0.09092	9.6377	0.013950	0.062722	0	1.21	0.2568

Figure 8: Predictive Mean Matching Parameter Estimates

Thus far only the results from multiple imputation methods have been used to analyze the dataset. Expectation Maximization can also be used to obtain good estimates, in fact, the estimates from the E-M algorithm approximate the maximum likelihood estimates. The default convergence criterion in SAS is $1 * 10^{-4}$, and the maximum number of iterations is 200. For this dataset the maximum number of iterations was reached before the convergence criterion was satisfied. *Figure 9* shows SAS output using the E-M algorithm. Many of the parameter estimates are radically different than those using multiple imputation methods. From *Figures 6* and *8* the intercept estimate was -1.30 and -1.50 respectively, while using the E-M algorithm the intercept estimate was 0.06. In addition to different estimates, all regression coefficients are significant in *Figure 9*, but this is an error. Using the E-M algorithm alone provides no way of quantifying variability for the parameter estimates, thus the standard errors and *p*-values displayed in *Figure 9* are artifacts from an invalid covariance matrix which SAS uses.

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	0.05966	0.01690	3.53	0.0004
FSW1	1	-0.00420	0.00075962	-5.53	<.0001
SSW1	1	-0.00755	0.00079591	-9.49	<.0001
ACE1	1	0.01016	0.00065174	15.60	<.0001
WNR1	1	0.01207	0.00033022	36.56	<.0001
UFE1	1	-0.00891	0.00027830	-32.01	<.0001
BPC1	1	0.10631	0.00238	44.74	<.0001
BPW1	1	-0.00440	0.00107	-4.12	<.0001
TPW1	1	0.00165	0.00052753	3.14	0.0017
ST11	1	0.01657	0.00207	8.02	<.0001
ST21	1	0.10167	0.00190	53.43	<.0001
ST31	1	0.04167	0.00186	22.37	<.0001
ST41	1	0.02300	0.00178	12.95	<.0001
ST51	1	0.18655	0.00113	165.08	<.0001

Figure 9: E-M Algorithm Estimates

Figure 10 shows a message from the SAS log, which warns about the *p*-values from *Figure 9*. In particular the messages warns that the *p*-values and other statistics that depend upon the number of observations should not be used in interpretations.

```
WARNING: The data set WORK.TENNISEM does not indicate how many observations were used to compute
         the COV matrix. The number of observations has been set to 10000. Statistics that depend
         on the number of observations (such as p-values) are not interpretable.
```

Figure 10: E-M Warning Message

Because Multiple Imputation yields accurate standard errors for imputed values and allows for the interpretation of *p*-values and confidence intervals, either Multiple Imputation method is preferred over the Expectation Maximization algorithm, under the assumption of MAR and multivariate normality. To keep this analysis simple, parametric methods were used, although the Propensity Score method is a valid Multiple Imputation method for MAR data with a Monotone Missing Pattern.

Code Used

To produce the results for the simulation and application, R, Excel VBA and SAS programming was used. Beginning with the simulation, the following lines of R code, create MCAR random, and estimates the bias, variance and mean square error.

```
library('HotDeckImputation')
#Creating the dataset
N <- 250
x1<-rnorm(N,6,1)
x2<-rnorm(N,0,1)
x3<-rnorm(N,-10,1)
y<--1+1*x1-1*x2+1*x3+rnorm(N,0,1)
#Making x3 MCAR
alpha=.3
r.x3.mcar<-rbinom(N,1,prob=alpha)
x3.mcar<-x3*(1-r.x3.mcar)+r.x3.mcar*99999
x3.mcar[x3.mcar==99999]=NA
#The complete case analysis
mm1 <- lm(y~x1+x2+x3.mcar, na.action=na.omit)
beta<-mm1$coeff
beta<- t(beta)
beta<- t(beta)
beta1 <- beta
std<-summary(mm1)$sigma
#The mean imputation analysis
design<- cbind(y,x1,x2,x3.mcar)
design<-data.frame(design)
x1.1<-design$x1[complete.cases(design)]
x2.1<-design$x2[complete.cases(design)]
x3.1<-design$x3.mcar[complete.cases(design)]
y.1<-design$y[complete.cases(design)]
x1.2<-design$x1[!complete.cases(design)]
x2.2<-design$x2[!complete.cases(design)]
y.2<-design$y[!complete.cases(design)]
x0.2<-rep(1,length(x2.2))
design<-as.matrix(design)
design<- impute.mean(design)
design<-data.frame(design)
mm1 <- lm(data=design,X1~X2+X3+X4)
beta<-mm1$coeff
beta<- t(beta)
beta<- t(beta)
beta2<- beta
#This plot illustrates reduction in variance of x3
#plot(design$X4, design$X1, xlab='Variable with Imputed Values',
ylab='Response')
#The Regresion Mean Analysis
matrix <- cbind(x0.2,x1.2,x2.2)
beta<-beta1[-4,]
beta <- t(beta)
beta <- t(beta)
x3.2 <- matrix %*% beta
matrix <- cbind(y.2,x1.2,x2.2,x3.2)
matrix0 <- cbind(y.1,x1.1,x2.1,x3.1)
matrix<- rbind(matrix0,matrix)
```

```
design<-matrix
design<-data.frame(design)
mm1<-lm(data=design,y.1~x1.1+x2.1+x3.1)
beta<-mm1$coeff
beta3<- beta
beta3<- t(beta3)
beta3<- t(beta3)
#This plot illustrates problems with Regression Mean Imputation
#plot(design$x3.1, design$x2.1, xlab='Imputed Variable',
ylab='Covariate')
#The stocahstic regression mean imputation
matrix <- cbind(x0.2,x1.2,x2.2)
z <- rnorm(length(x2.2),0,std)
beta<-beta1[-4,]
beta <- t(beta)
beta <- t(beta)
x3.2 <- matrix %*% beta + z
matrix <- cbind(y.2,x1.2,x2.2,x3.2)
matrix0 <- cbind(y.1,x1.1,x2.1,x3.1)
matrix<- rbind(matrix0,matrix)
design<-matrix
design<-data.frame(design)
mm1<-lm(data=design,y.1~x1.1+x2.1+x3.1)
beta<-mm1$coeff
beta4<- beta
beta4<- t(beta4)
beta4<- t(beta4)



for(i in 1:9999){
r.x3.mcar<-rbinom(N,1,prob=alpha)
x3.mcar<-x3*(1-r.x3.mcar)+r.x3.mcar*99999
x3.mcar[x3.mcar==99999]=NA
mm1 <- lm(y~x1+x2+x3.mcar, na.action=na.omit)
std<-summary(mm1)$sigma
beta <- mm1$coeff
beta <- t(beta)
beta <- t(beta)
beta1.loop <- beta
beta1 <- cbind(beta1,beta)
design<- cbind(y,x1,x2,x3.mcar)
design<-data.frame(design)
x1.1<-design$x1[complete.cases(design)]
x2.1<-design$x2[complete.cases(design)]
x3.1<-design$x3.mcar[complete.cases(design)]
y.1<-design$y[complete.cases(design)]
x1.2<-design$x1[!complete.cases(design)]
x2.2<-design$x2[!complete.cases(design)]
y.2<-design$y[!complete.cases(design)]
x0.2<-rep(1,length(x2.2))
design<-as.matrix(design)
design<- impute.mean(design)
design<-data.frame(design)
mm1 <- lm(data=design,X1~X2+X3+X4)
beta<-mm1$coeff
beta<- t(beta)
```

```
beta<- t(beta)
beta2<- cbind(beta2,beta)
matrix <- cbind(x0.2,x1.2,x2.2)
beta<-beta1.loop[-4,]
beta <- t(beta)
beta <- t(beta)
x3.2 <- matrix %*% beta
matrix <- cbind(y.2,x1.2,x2.2,x3.2)
matrix0 <- cbind(y.1,x1.1,x2.1,x3.1)
matrix<- rbind(matrix0,matrix)
design<-matrix
design<-data.frame(design)
mm1<-lm(data=design,y.1~x1.1+x2.1+x3.1)
beta<-mm1$coeff
beta<- t(beta)
beta<- t(beta)
beta3<- cbind(beta3,beta)
matrix <- cbind(x0.2,x1.2,x2.2)
z <- rnorm(length(x2.2),0,std)
beta<-beta1.loop[-4,]
beta <- t(beta)
beta <- t(beta)
x3.2 <- matrix %*% beta + z
matrix <- cbind(y.2,x1.2,x2.2,x3.2)
matrix0 <- cbind(y.1,x1.1,x2.1,x3.1)
matrix<- rbind(matrix0,matrix)
design<-matrix
design<-data.frame(design)
mm1<-lm(data=design,y.1~x1.1+x2.1+x3.1)
beta<-mm1$coeff
beta<-t(beta)
beta<-t(beta)
beta4<-cbind(beta4,beta)}

betahat01 <- mean(beta1[1,])
betahat11 <- mean(beta1[2,])
betahat21 <- mean(beta1[3,])
betahat31 <- mean(beta1[4,])
betahat02 <- mean(beta2[1,])
betahat12 <- mean(beta2[2,])
betahat22 <- mean(beta2[3,])
betahat32 <- mean(beta2[4,])
betahat03 <- mean(beta3[1,])
betahat13 <- mean(beta3[2,])
betahat23 <- mean(beta3[3,])
betahat33 <- mean(beta3[4,])
betahat04 <- mean(beta4[1,])
betahat14 <- mean(beta4[2,])
betahat24 <- mean(beta4[3,])
betahat34 <- mean(beta4[4,])
bias1 <- c(betahat01-1, betahat11 - 1, betahat21 +1, betahat31-1)
bias2 <- c(betahat02-1, betahat12 - 1, betahat22 +1, betahat32-1)
bias3 <- c(betahat03-1, betahat13 - 1, betahat23 +1, betahat33-1)
bias4 <- c(betahat04-1, betahat14 - 1, betahat24 +1, betahat34-1)
var1 <-
c(sd(beta1[1,]),sd(beta1[2,]),sd(beta1[3,]),sd(beta1[4,]))
```

```
var2 <-
c(sd(beta2[1,]),sd(beta2[2,]),sd(beta2[3,]),sd(beta2[4,]))
var3 <-
c(sd(beta3[1,]),sd(beta3[2,]),sd(beta3[3,]),sd(beta3[4,]))
var4 <-
c(sd(beta4[1,]),sd(beta4[2,]),sd(beta4[3,]),sd(beta4[4,]))
mse1 <- bias1^2+var1^2
mse2 <- bias2^2+var2^2
mse3 <- bias3^2+var3^2
mse4 <- bias4^2+var4^2
```

To analyze the tennis dataset in SAS the *.csv* files needed to be modified. Originally missing values were indicated by the characters NA. In order to read the data set in SAS, the NA characters needed to be deleted. A simple macro in Excel VBA can accomplish this task for each *.csv* file. The following line of code are the Excel VBA macro.

```
Sub DataCleanUp()

Dim rng As Range, cell As Range

Set rng = Range("A1:AP127")

For Each cell In rng
If cell.Value = "NA" Then
cell.Select
Selection.ClearContents
End If
Next cell

End Sub
```

After the *.csv* files were modified, the files were read into a SAS dataset. The Multiple Imputation methods used and E-M algorithm were all options for the SAS procedure, PROC MI. The regression coefficients for each multiply impute dataset were found using PROC REG, and the coefficients were pooled together using PROC MIANALYZE. *Figures 5, 6, 7, 8* are output from PROC MIANALYZE, while *Figure 9* is output from PROC REG. The following lines of code were used to accomplish this.

```
filename tennis1 'E:\Tennis\AusOpen-men-2013.csv';

data test26;
infile tennis1 DSD;
input Player1     Player2  Round    Result      FNL1  FNL2
FSP1  FSW1  SSP1  SSW1  ACE1  DBF1  WNR1
UFE1  BPC1  BPW1  NPA1  NPW1  TPW1  ST11
ST21  ST31  ST41  ST51  FSP2  FSW2  SSP2
SSW2  ACE2  DBF2  WNR2  UFE2  BPC2  BPW2
NPA2  NPW2  TPW2  ST12  ST22  ST32  ST42
ST52;

data tennis1;
      set work.test26;
      keep FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;
```

```
filename tennis1 'E:\Tennis\AusOpen-women-2013.csv';
data test27;
infile tennis1 DSD;
input Player1     Player2  Round    Result      FNL1  FNL2
FSP1  FSW1  SSP1  SSW1  ACE1  DBF1  WNR1
UFE1  BPC1  BPW1  NPA1  NPW1  TPW1  ST11
ST21  ST31  ST41  ST51  FSP2  FSW2  SSP2
SSW2  ACE2  DBF2  WNR2  UFE2  BPC2  BPW2
NPA2  NPW2  TPW2  ST12  ST22  ST32  ST42
ST52;

data tennis2;
      set work.test27;
      keep FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

filename tennis3 'E:\Tennis\FrenchOpen-men-2013.csv';
data test28;
infile tennis3 DSD;
input Player1     Player2  Round    Result      FNL1  FNL2
FSP1  FSW1  SSP1  SSW1  ACE1  DBF1  WNR1
UFE1  BPC1  BPW1  NPA1  NPW1  TPW1  ST11
ST21  ST31  ST41  ST51  FSP2  FSW2  SSP2
SSW2  ACE2  DBF2  WNR2  UFE2  BPC2  BPW2
NPA2  NPW2  TPW2  ST12  ST22  ST32  ST42
ST52;

data tennis3;
      set work.test28;
      keep FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

filename tennis4 'E:\Tennis\FrenchOpen-men-2013.csv';
data test29;
infile tennis4 DSD;
input Player1     Player2  Round    Result      FNL1  FNL2
FSP1  FSW1  SSP1  SSW1  ACE1  DBF1  WNR1
UFE1  BPC1  BPW1  NPA1  NPW1  TPW1  ST11
ST21  ST31  ST41  ST51  FSP2  FSW2  SSP2
SSW2  ACE2  DBF2  WNR2  UFE2  BPC2  BPW2
NPA2  NPW2  TPW2  ST12  ST22  ST32  ST42
ST52;

data tennis4;
      set work.test29;
      keep FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

proc append base=work.tennis1
             data=work.tennis2;
run;

proc append base=work.tennis1
             data=work.tennis3;
```

```
run;

proc append base=work.tennis1
             data=work.tennis4;
run;

data tennis0;
set tennis1;
if FNL1=. then delete;
if ACE1=. then delete;
run;

title'The Dataset';
proc print data=tennis0;
run;

title'Imputation via Monotone Regression Method';
proc mi data=work.tennis0
      seed=123456 out=outex1;
monotone reg(ST21 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1
ST11);
monotone reg(ST31 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1 ST11
ST21);
monotone reg(ST41 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1 ST11
ST21 ST31);
monotone reg(ST51 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1 ST11
ST21 ST31 ST41);
var FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

proc reg data=outex1 outest=outreg1 covout noprint;
   model FNL1= FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
   by _Imputation_;
run;

title'Pooling the Parameter Estimates';
proc mianalyze data=outreg1 edf=422;
   modeleffects Intercept FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

title'Imputation via Monotone Predictive Mean Matching Method';
proc mi data=work.tennis0
      seed=123456 out=outex2;
monotone regpmm(ST21 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1
ST11);
monotone regpmm(ST31 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1
ST11 ST21);
monotone regpmm(ST41 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1
ST11 ST21 ST31);
monotone regpmm(ST51 = FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1 TPW1
ST11 ST21 ST31 ST41);
var FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;
```

```
proc reg data=outex2 outest=outreg2 covout noprint;
   model FNL1= FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
   by _Imputation_;
run;

title'Pooling the Parameter Estimates';
proc mianalyze data=outreg2 edf=422;
   modeleffects Intercept FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;

title'Using the E-M Alg';
proc mi data=tennis0 nimpute=0;
em outem=tennisem;
var FNL1 FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;


proc reg data=tennisem outest=outreg2 covout;
   model FNL1= FSW1 SSW1 ACE1 WNR1 UFE1 BPC1 BPW1
      TPW1 ST11 ST21 ST31 ST41 ST51;
run;
```

References

Enders, C. (2010). Applied missing data analysis. New York: Guilford Press.

Imputation (statistics). (2015, October 31). Retrieved December 4, 2015, from https://en.wikipedia.org/wiki/Imputation_(statistics)

Missing data. (2015, October 30). Retrieved December 4, 2015, from https://en.wikipedia.org/wiki/Missing_data

SAS/STAT(R) 9.3 User's Guide. (2009, September 1). Retrieved December 4, 2015, from http://support.sas.com/documentation/cdl/en/statug/63962/HTML/default/viewer.htm#mi_toc.htm

UCI Machine Learning Repository: Data Set. (2014, June 1). Retrieved December 4, 2015, from https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics