Appendix A: Study Details

1. Derivation of Equation 5

We first begin by calculating the prediction and its variance using the predict then combine approach (PC). Let $\widehat{\mathbf{B}}$ be a *k* x *m* matrix giving estimates for the *k* predictors of the analysis model in each of the *m* imputed data sets. Linear predictions can be obtained by pre-multiplying $\widehat{\mathbf{B}}$ by $\boldsymbol{s}$, a 1 x *k* vector of first derivatives of the linear transformation function with respect to each predictor.

$$\hat{p} = \boldsymbol{s}\widehat{\mathbf{B}}$$

The combined prediction is the expected value of the *m* individual predictions.

$$\bar{p} = \mathrm{E}\left(\boldsymbol{s}\widehat{\mathbf{B}}\right)$$

which is a 1 x *k* vector with the expected value listed in each of the *k* positions. The within variance of the combined prediction is:

$$\mathrm{V_W} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{s}\widehat{\mathbf{V}}_j\boldsymbol{s}'$$

where $\widehat{\mathbf{V}}_j$ is the *k* x *k* covariance matrix from the model estimated using imputed data set *j*, *j* = 1, . . . , *m*. The between variance is the variance of the *m* predictions.

$$\mathrm{V_B} = \frac{1}{m-1}\sum_{j=1}^{m} (\hat{p} - \bar{p})^{\mathbf{2}}$$

$$\mathrm{V_B} = \frac{1}{m-1}\sum_{j=1}^{m} \left(\boldsymbol{s}\widehat{\mathbf{B}} - \mathrm{E}\left(\boldsymbol{s}\widehat{\mathbf{B}}\right)\right)^{\mathbf{2}}$$

$$\mathrm{V_B} = \frac{1}{m-1}\left(\boldsymbol{s}\widehat{\mathbf{B}} - \mathrm{E}\left(\boldsymbol{s}\widehat{\mathbf{B}}\right)\right)\left(\boldsymbol{s}\widehat{\mathbf{B}} - \mathrm{E}\left(\boldsymbol{s}\widehat{\mathbf{B}}\right)\right)'$$

Because the contents of $\boldsymbol{s}$ are fixed, we can treat them as constants and rearrange:

$$\mathrm{V_B} = \frac{1}{m-1}\left[\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)\right]\left[\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)\right]'$$

$$\mathrm{V_B} = \frac{1}{m-1}\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)'\boldsymbol{s}'$$

where $\mathrm{E}(\widehat{\mathbf{B}})$ is a *k* x *m* matrix with each row listing the mean of one of the *k* predictors in each of the *m* columns. The total variance of the prediction using PC, then, is:

$$\mathrm{V}_{\mathrm{PC}} = \mathrm{V}_{\mathrm{W}} + \mathrm{V}_{\mathrm{B}}\left(1 + \frac{1}{m}\right)$$

$$\mathrm{V}_{\mathrm{PC}} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{s}\widehat{\mathbf{V}}_j\boldsymbol{s}' + \frac{m+1}{m(m-1)}\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)'\boldsymbol{s}' \quad \text{(A1)}$$

Predictions can also be calculated directly from the combined model. The covariance matrix of the combined model is:

$$\widehat{\mathbf{V}}_{\mathrm{C}} = \widehat{\mathbf{V}}_{\mathrm{W}} + \widehat{\mathbf{V}}_{\mathrm{B}}\left(1 + \frac{1}{m}\right) \quad \text{(A2)}$$

where $\widehat{\mathbf{V}}_{\mathrm{W}}$ and $\widehat{\mathbf{V}}_{\mathrm{B}}$ are defined as:

$$\widehat{\mathbf{V}}_{\mathrm{W}} = \frac{1}{m}\sum_{j=1}^{m}\widehat{\mathbf{V}}_j \quad \text{(A3)}$$

$$\widehat{\mathbf{V}}_{\mathrm{B}} = +\frac{1}{m-1}\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)' \quad \text{(A4)}$$

The variance of the prediction can be calculated using the delta method.

$$\mathrm{V}_{\bar{p}} = \boldsymbol{s}\widehat{\mathbf{V}}_{\mathrm{C}}\boldsymbol{s}'$$

We can substitute in equation A2, then A3 and A4 and rearrange:

$$\mathrm{V}_{\mathrm{C}} = \boldsymbol{s}\left[\widehat{\mathbf{V}}_{\mathrm{W}} + \widehat{\mathbf{V}}_{\mathrm{B}}\left(1 + \frac{1}{m}\right)\right]\boldsymbol{s}'$$

$$\mathrm{V}_{\mathrm{C}} = \boldsymbol{s}\left[\frac{1}{m}\sum_{j=1}^{m}\widehat{\mathbf{V}}_j + \frac{1}{m-1}\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)'\left(1 + \frac{1}{m}\right)\right]\boldsymbol{s}'$$

$$\mathrm{V}_{\mathrm{C}} = \boldsymbol{s}\left[\frac{1}{m}\sum_{j=1}^{m}\widehat{\mathbf{V}}_j\right]\boldsymbol{s}' + \boldsymbol{s}\left[\frac{1}{m-1}\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)'\left(1 + \frac{1}{m}\right)\right]\boldsymbol{s}'$$

$$\mathrm{V}_{\mathrm{C}} = \boldsymbol{s}\left[\frac{1}{m}\sum_{j=1}^{m}\widehat{\mathbf{V}}_j\right]\boldsymbol{s}' + \frac{m+1}{m(m-1)}\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)\left(\widehat{\mathbf{B}} - \mathrm{E}(\widehat{\mathbf{B}})\right)'\boldsymbol{s}' \quad \text{(A5)}$$

Given the rules for linear transformations,

$$\boldsymbol{s}\left[\frac{1}{m}\sum_{j=1}^{m}\widehat{\mathbf{V}}_j\right]\boldsymbol{s}' = \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{s}\widehat{\mathbf{V}}_j\boldsymbol{s}'$$

Substituting this result into equation A5 produces

$$\mathrm{V_C} = \frac{1}{m}\sum_{j=1}^{m}\boldsymbol{s}\widehat{\mathbf{V}}_j\boldsymbol{s}' + \frac{m+1}{m(m-1)}\boldsymbol{s}\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)\left(\widehat{\mathbf{B}} - \mathrm{E}\left(\widehat{\mathbf{B}}\right)\right)'\boldsymbol{s}' \qquad \text{(A6)}$$

which is equivalent to equation A1, the variance of the prediction obtained using PC.

2. Coding Details for Empirical Illustration

All data were taken from the pooled GSS data file and were used without recoding. Respondents' and fathers' occupational prestige scores ranged from 17 to 86, where higher values represent more prestigious occupations (GSS variables PRESTG80 and PAPRES80). Highest degrees earned for both respondents and fathers were included in analyses as mutually exclusive indicators for the following categories: less than high school, high school, associate/junior college, bachelor's, and graduate (GSS variables DEGREE and PADEG). All models were weighted using the GSS's constructed weights, wtssall.

3. Models Used for Tables 1 and 2

	Table 1			Table 2		
	Estimate	**S.E.**		**Estimate**	**S.E.**	
Respondent's education (ref = < high school)						
high school	5.84	(0.19)	***	5.84	(0.19)	***
junior college	12.04	(0.34)	***	12.04	(0.34)	***
BA	17.55	(0.27)	***	17.55	(0.27)	***
graduate degree	25.99	(0.33)	***	25.99	(0.33)	***
Father's education (ref = < high school)						
high school	-0.01	(0.18)		-0.01	(0.18)	
junior college	-0.46	(0.47)		-0.47	(0.47)	
BA	-0.27	(0.31)		-0.28	(0.31)	
graduate degree	-0.63	(0.38)		-0.64	(0.37)	
Father's occupational prestige	0.07	(0.01)	***	0.06	(0.03)	*
Year (ref = 1988)						

1989	0.23	(0.45)		-0.42	(1.73)	
1990	-0.25	(0.46)		-0.18	(1.81)	
1991	0.04	(0.44)		-0.21	(1.74)	
1993	-0.34	(0.43)		-0.36	(1.66)	
1994	-0.55	(0.39)		-1.71	(1.53)	
1996	-0.57	(0.39)		-1.08	(1.55)	
1998	0.29	(0.40)		0.95	(1.58)	
2000	0.27	(0.40)		-0.02	(1.63)	
2002	-0.19	(0.40)		0.05	(1.61)	
2004	0.54	(0.40)		-0.80	(1.58)	
2006	-0.03	(0.38)		0.00	(1.54)	
2008	-0.72	(0.44)		-0.68	(1.70)	
2010	-1.02	(0.44)	*	-1.76	(1.71)	
Father's prestige X Year (ref = 1988)						
1989				0.016	(0.04)	
1990				-0.001	(0.04)	
1991				0.006	(0.04)	
1993				0.001	(0.04)	
1994				0.027	(0.04)	
1996				0.012	(0.04)	
1998				-0.015	(0.04)	
2000				0.007	(0.04)	
2002				-0.005	(0.04)	
2004				0.031	(0.04)	
2006				-0.001	(0.04)	
2008				-0.001	(0.04)	
2010				0.017	(0.04)	
Intercept	31.69	(0.43)	***	31.983	(1.26)	***

Note: * p < 0.05; ** p < 0.01; *** p < 0.001

4. Simulation Details

Simulations compared the mean squared error of both the predict-then-combine (PC) and combine-then-predict (CP) methods for data with rates of missingness ranging from 10-50% of each variable. Results for each level of missingness are based on 1000 repetitions of the following steps.

First, a random data matrix was created that included 3 predictors variables (x_1-x_3, inter-correlated at 0.5), and 3 outcome variables (one for each type of model). Outcome variables were generated using the following equations:

Logit	$y^*_{logit} = 0.5x_1 + 0.5x_2 + 0.5x_3 + e_{logit}$ $y_{logit} = \begin{cases} 1 \text{ if } y^*_{logit} > 0 \\ 0 \text{ otherwise} \end{cases}$
Probit	$y^*_{probit} = 0.5x_1 + 0.5x_2 + 0.5x_3 + e_{probit}$ $y_{probit} = \begin{cases} 1 \text{ if } y^*_{probit} > 0 \\ 0 \text{ otherwise} \end{cases}$
Poisson	$\lambda = e^{(0.5x_1+0.5x_2+0.5x_3)}$ $y_{poisson} \sim \text{Poisson}(\lambda)$

where e_{logit} was randomly generated from a logistic distribution such that $e_{logit} \sim$ Logistic(0,1), e_{probit} was randomly generated from a normal distribution such that $e_{probit} \sim$ N(0,1), and $y_{poisson}$ was randomly generated from a Poisson distribution with mean λ, as indicated.

The specified proportion of values was then randomly sampled from each variable and set to missing (missing completely at random), and the data matrix was then imputed using the expectation maximization bootstrap approach outlined by Honaker and King (2010). The number of imputations was set to match the proportion of missing values (e.g., m = 30 with 30% missing, see White et al. 2011, section 7.3). Cases originally missing data on the outcome were used during imputation but excluded prior to analyses (von Hippel 2007).

Logit, probit, and Poisson models were then estimated in each imputed data set using the outcomes listed in the table above and all three predictor variables. Finally, the CP and PC methods were used to calculate non-linear predictions appropriate to the models: probabilities for logit and probit models, and counts for the Poisson model. These predictions set all *x* variables to 1.

This resulted in 1000 predictions for both PC and CP. The mean squared error was calculated by taking the mean of the squared difference between the estimated and the true prediction for each model. In each case, the true linear prediction was 0.5*1 + 0.5*1 + 0.5*1 = 1.5 (by construction), leading to non-linear predictions as shown below.

Logit	$p_{logit} = \frac{e^{1.5}}{1 + e^{1.5}} \approx 0.82$
Probit	$p_{probit} = \Phi(1.5) \approx 0.93$
Poisson	$p_{poisson} = e^{1.5} \approx 4.48$

R code for the simulation is available on the author's website at www.andrewamiles.com.

REFERENCES

Von Hippel, Paul T. 2007. "Regression with Missing Ys: An Improved Strategy for Analyzing Multiply Imputed Data." *Sociological Methodology* 37(1):83–117.

Honaker, James, and Gary King. 2010. "What to Do about Missing Values in Time-Series Cross-Section Data." *American Journal of Political Science* 54(2):561–81.

White, Ian R., Patrick Royston, and Angela M. Wood. 2011. "Multiple Imputation Using Chained Equations: Issues and Guidance for Practice." *Statistics in Medicine* 30(4):377–99.

Appendix B: **Sample Code for Implementing Predict then Combine and Combine then Predict Methods in Stata and R**

Stata has a number of easy-to-use commands for producing predictions from MI data (e.g., mi predict, mi predictnl, and the user written mimrgns), but all use the predict then combine approach (PC). In R PC methods can be implemented using list and matrix functions (e.g., sapply, rowMeans), or using the functions available in the mitools package. In both R and Stata, combine then predict methods (CP) currently require extracting and working directly with the coefficients and covariance matrix of the combined model. Hopefully the CP approach will be implemented in future software releases, making it as easy to use as PC.

Until that occurs, I have included sample code for both approaches below. The examples given for both Stata and R first generate data, randomly set some data to missing, impute the data, then apply the predict then combine (PC) and combine then predict (CP) techniques to calculate a predicted value and its standard error. The examples are designed to take little computational time using either method. The Stata code requires the user written command 'mimrgns,' and the R code requires the packages 'Amelia' and 'mitools.'

Stata code

```
/*generate data*/
set obs 1000
matrix mu=1,0,0,0
matrix sigma=(1, .5, .5, .5 \ .5, 1, .5, .5 \ .5, .5, 1, .5 \ ///
      .5, .5, .5, 1)
drawnorm x1 x2 x3 x4, cov(sigma) means(mu)
gen y = 1 + .2*x1 + .3*x2 + .4*x3 + rnormal(0,2)

/*introduce missingness, about 10% for each variable*/
gen p = .1
foreach var of varlist x1-y {
      replace `var' = . if uniform() < p
}

/*generate imputations*/
mi set wide
mi register imputed x1-y
mi impute mvn x1-y, add(10)

/*estimate pooled model*/
gen ymiss=missing(y)
mi estimate, dots post: regress y x1 x2 x3 if ymiss != 1

/***estimate predictions for set covariate values***/

/* Predict then Combine (PC) method */
mimrgns, at(x1=2 x2=1 x3=1)
```

```
/* Combine then Predict (CP) method */
matrix s = 2,1,1,1            /*first derivatives of prediction equation*/
matrix phat = s*e(b)'         /*prediction*/
matrix phat_var = s*e(V)*s'   /*variance of prediction*/
matrix list phat
di sqrt(phat_var[1,1])        /*SE of prediction*/
```

R code

```
#load required packages
library(Amelia); library(mitools)

#generate data
n=1000
mu=c(1,0,0,0)
sigma=matrix(.5,nrow=4, ncol=4)
diag(sigma)=1
dat=data.frame(mvrnorm(n, mu, sigma))
colnames(dat)=c("x1", "x2", "x3", "x4")
dat$y = with(dat, 1 + .2*x1 + .3*x2 + .4*x3 + rnorm(n,0,2))

#introduce missingness
rowmiss=sample(1:n, n/2, replace=T)
colmiss=sample(1:ncol(dat), n/2, replace=T)
for(i in 1:length(rowmiss)){
      dat[rowmiss[i], colmiss[i]]=NA
}

#generate imputations
dat.imp=amelia(dat, m=10)

#estimate a final (combined) MI model
milist=imputationList(dat.imp$imputations)
ymiss=is.na(dat$y)
impmods=with(milist, lm(y ~ x1 + x2 + x3, subset=ymiss==F))
mod=MIcombine(impmods)

#### estimate predictions for set covariate values ####

#Predict then Combine (PC) method
preds=sapply(impmods, predict, newdata=data.frame(x1=2, x2=1, x3=1),
se.fit=T)
phats=preds[1,]
Vw=(unlist(preds[2,]))^2   #square the SE to get within imputation variance
MIcombine(phats, Vw)

#Combine then Predict (CP) method
```

```
s=matrix(c(1,2,1,1), nrow=1)    #first derivatives of prediction equation
s %*% coef(mod)    #prediction
sqrt(s %*% vcov(mod) %*% t(s))    #SE

#if it is not obvious to you how to obtain the s vector, write out the
prediction equation, then use the deriv() function
form= formula("~ intercept*1 + x1*2 + x2*1 + x3*1")
deriv(form, namevec=c("intercept", "x1", "x2", "x3"))
```