Multiple Imputation

Paul E. Johnson[1] [2]

[1]Department of Political Science

[2]Center for Research Methods and Data Analysis, University of Kansas

2013

Overview

- Why Impute?
- Amelia
- mice
- mi
- aregImpute

Listwise Deletion

- In almost all software, "listwise deletion" has been the default. If any variable is missing, a case is completely dropped from the analysis.
- Most people seem to agree that is bad–"biased parameter estimates"
- Until recently, practitioners ignored the problem.
- Research on this has been steadily accumulating since 1970s
- Statistical researchers are trying to find a more-or-less convenient, "idiot proof" procedure

Rubin proposed Multiple Imputation

- Jargon: MAR ("Missing at Random") means that the chance of a missing score is predictable using information in other variables (and not predictable by other unmeasured forces)
- Rubin's proposal
  1. Use many variables, including the dependent variable and variables not planned for inclusion in the final model, to predict missings
  2. Create several "Imputed" data sets.
  3. Run Each analysis on Each Imputed Dataset
  4. Combine the estimates, weight them to take uncertainty into account.

Do You Do It Yourself?

- Rubin suggested the imputations could be done at a data center when they supply the dataset. MI would be done "once and for all," and the imputed missings would be distributed as one collection.
- That approach was impractical for a number of reasons.
- Many routines to impute missing values have been proposed.
- This research area is still under active development
- Caution: I'm not an MI authority (just a guy demonstrating some R packages)

What are we looking for?

- What format does a routine expect from our data?
- Are the imputations returned in a manageable format?
- Is it difficult to conduct the analysis on each separate dataset?
- How to best pool the estimates together and summarize them?

Points of Concern

- Calculation of "imputation averaged" results
  - Good theory/method exists for MLE of "slope coefficients".
- "Rubin's Rules" for slope & variance estimates
  - For slope estimates, average the imputed, $\hat{\beta} = \sum_{i=1}^{m} \hat{\beta}_j$
  - Variance estimate for $\hat{\beta}$ combines
    1. average of $\widehat{Var(\hat{\beta}_j)}$, $\sum_{i=1}^{m} = \widehat{Var(\hat{\beta}_j)}$, plus
    2. a penalty for uncertainty between $\hat{\beta}_j$, $\frac{1}{1+m} \sum (\hat{\beta}_j - \hat{\beta})^2$.
- Less good theory/tools on other statistics (R^2, deviance, etc.)
- Difficult choices about "openness" and "interoperability" with other R functions
- Caution about terminology: imputation sometimes means
  - The candidates to "fill in" for NAs
  - A completed data frame with the NAs are replaced by the candidates

Amelia

King, Gary, James Honaker, Anne Joseph, and Kenneth Scheve. 2001. "Analyzing Incomplete Political Science Data: An Alternative Algorithm for Multiple Imputation." The American Political Science Review 95(1): 49-69.

James Honaker, Gary King, Matthew Blackwell (2011). Amelia II: A Program for Missing Data. Journal of Statistical Software, 45(7), 1-47. URL http://www.jstatsoft.org/v45/i07/.

Rough Sketch of Amelia

- Assume all variables are drawn from one Multivariate Normal Distribution, $MVN(\mu, \Sigma)$
- Conduct series of complicated algorithms to estimate μ and Σ
- After estimating μ and Σ, then draw random samples from the MVN to fill in missing values
- Basic idea similar to "Norm" (J. Schafer), but algorithm may be faster (EM with "importance sampling")

Interface

```
amelia(x, m = 5, p2s = 1,frontend = FALSE, idvars =
    NULL, ts = NULL, cs = NULL, polytime = NULL,
   splinetime = NULL, intercs = FALSE, lags = NULL
   , leads = NULL, startvals = 0, tolerance = 0
   .0001, logs = NULL, sqrts = NULL, lgstc = NULL,
    noms = NULL, ords = NULL, incheck = TRUE,
   collect = FALSE, arglist = NULL, empri = NULL,
   priors = NULL, autopri = 0.05, emburn = c(0,0),
    bounds = NULL, max.resample = 100, ...)
```

Note: amelia uses all of the supplied variables in imputations except vars declared as "idvars." To save memory, one should remove all extraneous variables first, rather than use the "idvars" feature to ask amelia to ignore them.

Surprising, Possibly True

- Most people say "but my variables are not Normal." (gender, survey scales, etc)
- King (and others) argue the approximation is not harmful (various reasons)
- Amelia allows user to specify variables as "nominal" and "ordinal"
  - Nominal variables: The normal imputations are "rounded off" to values in the observed scale {0,1,2}
  - Ordinal variables: Optionally "rounded off" to integers, but instructions discourage that
  - They suggest a 7 point scale might meaningfully have imputed values in-between the integers

Grab Some Data, Impose Some Missings

Thanks to Chuck Cleland who suggested this example in r-help

```
options(digits=2)
if (!file.exists("examples")) dir.create("examples
    ")
if (!file.exists("examples/titanic.txt"))
 download.file("http://lib.stat.cmu.edu/S/Harrell/
     data/ascii/titanic.txt", "examples/
     titanic.txt")
titanic <- read.table("examples/titanic.txt", sep
    = ',', header = TRUE)
titanic0 <- titanic
save(titanic0, file="examples/titanic0.rda")
set.seed(4321)
titanic$sex[sample(nrow(titanic), 10)] <- NA
titanic$pclass[sample(nrow(titanic), 10)] <- NA
titanic$survived[sample(nrow(titanic), 10)] <- NA
```

The "Most Complete" Version of the Data Says ...

```
fullglm <- glm(survived ~ pclass + sex + age,
    family = binomial, data = titanic0)
library(xtable)
tout <- xtable(fullglm)
print(tout, type = "latex")
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	4.5222	0.4710	9.60	0.0000
pclass2nd	-1.4952	0.2820	-5.30	0.0000
pclass3rd	-2.8413	0.3389	-8.38	0.0000
sexmale	-3.0867	0.2411	-12.80	0.0000
age	-0.0493	0.0087	-5.65	0.0000

After Imposing some more Missings, The ListWise Deletion Results

```
ldglm <- glm(survived ~ pclass + sex + age, family
    = binomial, data = titanic)
library(xtable)
tout <- xtable(ldglm)
print(tout, type = "latex")
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	4.3706	0.4709	9.28	0.0000
pclass2nd	-1.4617	0.2847	-5.13	0.0000
pclass3rd	-2.7442	0.3403	-8.06	0.0000
sexmale	-3.0773	0.2414	-12.75	0.0000
age	-0.0464	0.0088	-5.30	0.0000

Use Amelia to create Imputed Data Sets

```
library(Amelia) # generate multiple imputations
titanic.amelia <- amelia(subset(titanic, select =
    c('survived', 'pclass', 'sex', 'age', '
    embarked')), m = 10,  noms = c('survived', '
    pclass', 'sex', 'embarked'),
 emburn = c(500, 500), p2s = F)
```

p2s=F turns off screen output that overflows the presentation software

Note: Now use tools not from Amelia to Analyze the Data and Summarize it

- Lets try to use general purpose tools to estimate and summarize these models.
- The imputations are in an R list, so the general "lapply" function can be used to fit any kind of model that can accept a data frame as an argument.
- The R package mitools (Thomas Lumley) has tools to combine estimates of slopes and calcuate the Rubin-adjusted standard errors.

lapply Conducts the glm for Each Imputed Set

```
allimplogreg <- lapply(titanic.amelia$imputations,
     function(x){
      glm(survived ~ pclass + sex + age, family =
          binomial, data = x)
 })
```

Post Processing with "mitools"

```
options(digits=2)
library(mitools) # MIextract
betas <- MIextract(allimplogreg, fun = coef)
vars <- MIextract(allimplogreg, fun = vcov)
summary(MIcombine(betas, vars))
```

```
Multiple imputation results:
      MIcombine.default(betas, vars)
            results    se (lower upper) missInfo
(Intercept)    4.03 0.532  2.926  5.128      66 %
pclass2nd     -1.44 0.258 -1.951 -0.931      27 %
pclass3rd     -2.71 0.308 -3.339 -2.081      57 %
sexmale       -2.53 0.175 -2.872 -2.184      18 %
age           -0.05 0.011 -0.073 -0.027      73 %
```

mi.inference from mix offers effective df and fm

```
library(mix)
se.glm <- MIextract(allimplogreg, fun = function(x){sqrt(diag(vcov(
    x)))})
as.data.frame(mi.inference(betas, se.glm))
```

	est	std.err	df	signif	lower	upper	r	fminf
(Intercept)	4.03	0.532	23	1.2e−07	2.926	5.128	1.72	0.66
pclass2nd	−1.44	0.258	129	1.3e−07	−1.951	−0.931	0.36	0.27
pclass3rd	−2.71	0.308	31	6.3e−10	−3.339	−2.081	1.17	0.57
sexmale	−2.53	0.175	300	0.0e+00	−2.872	−2.184	0.21	0.18
age	−0.05	0.011	18	2.5e−04	−0.073	−0.027	2.39	0.73

df: degrees of freedom associated with the t reference distribution used for interval estimates.
r: estimated relative increases in variance due to nonresponse.
fminf: estimated fractions of missing information.

Compare Side-by-Side: MI and LD results

```
df1 <- as.data.frame(mi.inference(betas, se.glm))
df2 <- cbind(df1[,1:2], ldbeta = coef(ldglm), ldse = sqrt(diag(vcov
    (ldglm))))
df2
```

	est	std.err	ldbeta	ldse
(Intercept)	4.03	0.532	4.371	0.4709
pclass2nd	−1.44	0.258	−1.462	0.2847
pclass3rd	−2.71	0.308	−2.744	0.3403
sexmale	−2.53	0.175	−3.077	0.2414
age	−0.05	0.011	−0.046	0.0088

Make the Missings Worse!

```
set.seed(234234)
titanic <- titanic0
titanic$sex[sample(nrow(titanic), 400)] <- NA
titanic$pclass[sample(nrow(titanic), 400)] <- NA
titanic$survived[sample(nrow(titanic), 400)] <- NA
```

Then estimate

- new "ldglm" (listwise deletion estimate)
- 10 fresh imputed datasets and regressions for each

New Listwise Deletion Model ldglm

```
ldglm <- glm(survived ~ pclass + sex + age, family
    =binomial, data = titanic)
library(xtable)
tout <- xtable(ldglm)
print(tout, type = "latex")
```

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	5.9001	0.9254	6.38	0.0000
pclass2nd	-2.1114	0.5233	-4.03	0.0001
pclass3rd	-4.4776	0.7632	-5.87	0.0000
sexmale	-3.6943	0.5480	-6.74	0.0000
age	-0.0657	0.0162	-4.07	0.0000

MI summary estimates

```
summary(MIcombine(betas, vars))
```

```
Multiple imputation results:
      MIcombine.default(betas, vars)
             results      se (lower upper) missInfo
(Intercept)    2.89  0.4254   2.011  3.770      65 %
pclass2nd     -0.91  0.2682  -1.453 -0.364      53 %
pclass3rd     -2.10  0.2799  -2.679 -1.526      63 %
sexmale       -1.73  0.2213  -2.184 -1.273      63 %
age           -0.04  0.0083  -0.057 -0.023      65 %
```

Compare Side-by-Side: MI and LD results

```
df2.1 <- as.data.frame(mi.inference(betas, se.glm))
df2.2 <- cbind("MI",df2.1[,1:2], "LD", ldbeta = coef(ldglm), ldse =
      sqrt(diag(vcov(ldglm))), "full", est = coef(fullglm))
df2.2
```

	"MI"	est	std.err	"LD"	ldbeta	ldse	"full"	est
(Intercept)	MI	2.89	0.4254	LD	5.900	0.925	full	4.522
pclass2nd	MI	−0.91	0.2682	LD	−2.111	0.523	full	−1.495
pclass3rd	MI	−2.10	0.2799	LD	−4.478	0.763	full	−2.841
sexmale	MI	−1.73	0.2213	LD	−3.694	0.548	full	−3.087
age	MI	−0.04	0.0083	LD	−0.066	0.016	full	−0.049

Maybe confidence intervals help

	est	lower	upper	"full"	beta	2.5 %	97.5 %
(Intercept)	2.89	2.011	3.770	full	4.522	3.632	5.480
pclass2nd	−0.91	−1.453	−0.364	full	−1.495	−2.059	−0.952
pclass3rd	−2.10	−2.679	−1.526	full	−2.841	−3.527	−2.196
sexmale	−1.73	−2.184	−1.273	full	−3.087	−3.575	−2.628
age	−0.04	−0.057	−0.023	full	−0.049	−0.067	−0.033

	LDbeta	2.5 %	97.5 %
(Intercept)	5.900	4.218	7.866
pclass2nd	−2.111	−3.192	−1.128
pclass3rd	−4.478	−6.114	−3.101
sexmale	−3.694	−4.879	−2.707
age	−0.066	−0.099	−0.036

Multiple Imputation via Chained Equations

Stef van Buuren, Karin Groothuis-Oudshoorn (2011). MICE: Multivariate Imputation by Chained Equations in R. Journal of Statistical Software, 45(3): 1-67.

Stef van Buuren (2012). Flexible Imputation of Missing Data. Boca Raton, FL: Chapman & Hall/CRC Press.

Rough Sketch

- Strategy quite different from Amelia and other MVN based theories
- MICE: separately process each column, predicting it from all the others.
  "The algorithm imputes an incomplete column (the target column) by generating 'plausible' synthetic values given other columns in the data."
- Cycle through columns over and over, until model converges (in MCMC sense), then draw samples to impute.

Recommends "predictive mean matching" to select imputed values

- When filling in missings, find cases with similar predicted values to the case in question
- From among those cases, collect their list of actual observed scores
- Draw imputations from that subset of actual scores
- "Automatically" solves the problem that imputations might have impossible values
  - Imputations for categorical variables always match the original scale (sex is always 0 or 1, never 0.64)
  - When a variable is badly skewed, the PMM always selects a realistic value.

The mice Interface

```
mice(data, m = 5, method = vector("character",
    length=ncol(data)), predictorMatrix = (1 -
    diag(1, ncol(data))), visitSequence = (1:ncol
    (data))[apply(is.na(data), 2, any)], post =
    vector("character", length = ncol(data)),
    defaultMethod = c("pmm", "logreg", "polyreg",
     "polr"), maxit = 5, diagnostics = TRUE,
    printFlag = TRUE, seed = NA, ...
    )
```

Special mice features

- "fine grained" management of imputation algorithms for different types of data
- Defaults:

data type	default	also available
numeric	pmm (predictive mean matching)	norm, 2level
binary	logreg (logistic regression)	lda
factor	polyreg (Bayesian polytomous regression)	
factor: ordinal	polr (prop. odds logistic (MASS))	

- Possible to
  - add user-defined predictive tools
  - control the sequence of column processing

Other Handy mice Features

- complete: function can
  - return any of the individual imputed data frames
  - return all data frames combined in the "long" format (rows stacked together)
  - return all frames combined in the "wide" format (columns side-by-side)
- pool: outputs many of Rubin's suggested diagnostic formulae (param, var, R^2)
- summary(pool( )): distills parameter estimates

nhanes: small test data frame supplied with mice

```
library(mice)
nhanes
```

```
   age bmi hyp chl
1    1  NA  NA  NA
2    2  23   1 187
3    1  NA   1 187
4    3  NA  NA  NA
5    1  20   1 113
6    3  NA  NA 184
7    1  22   1 118
8    1  30   1 187
9    2  22   1 238
10   2  NA  NA  NA
11   1  NA  NA  NA
12   2  NA  NA  NA
13   3  22   1 206
14   2  29   2 204
15   1  30   1  NA
16   1  NA  NA  NA
17   3  27   2 284
18   2  26   2 199
19   1  35   1 218
20   3  26   2  NA
21   1  NA  NA  NA
```

nhanes: small test data frame supplied with mice ...

```
22    1   33    1  229
23    1   28    1  131
24    3   25    1   NA
25    2   27    1  186
```

Test That Out

```
imp <- mice(nhanes, printFlag = FALSE)
fit <- with(data = imp, exp = lm(bmi ~ hyp + chl))
fitpool <- pool(fit)
fitpool
```

```
Call: pool(object = fit)

Pooled coefficients:
(Intercept)         hyp         chl
     20.592      -1.176       0.037

Fraction of information about the coefficients
    missing due to nonresponse:
(Intercept)         hyp         chl
       0.41        0.17        0.25
```

What's all that?

Inside the outpute object from pool, there is a wealth of information that previous editions of mi did report automatically. That structure includes

```
fit <- list(call = call, call1 = object$call,
    call2 = object$call1, nmis = object$nmis, m =
    m, qhat = qhat, u = u, qbar = qbar, ubar =
    ubar, b = b, t = t, r = r, dfcom = dfcom, df =
     df,          fmi = fmi, lambda = lambda)
```

qhat: matrix of m complete data fits	b: within imputation variance
r: rel. incr var due to nonresponse	t: total variance of pooled estimates
qbar: pooled estimate	u: Variance matrices from m fits ($var \times var \times m$)
ubar: mean of variances across m fits	gamma: prop. variance explained by imputations
dfcom: df in complete analysis	df: df for pooled estimates
	fmi: fraction missing information

summary of pooled fit

```
round(summary(pool(fit))[,c(1:4,6:9)],2)
```

	est	se	t	df	lo 95	hi 95	nmis	fmi
(Intercept)	20.59	5.07	4.06	11	9.38	31.81	NA	0.41
hyp	−1.18	2.21	−0.53	18	−5.81	3.46	8	0.17
chl	0.04	0.02	1.50	15	−0.02	0.09	10	0.25

It Gracefully Handles Factor Variables

```
nhanesf <- nhanes
nhanesf$age <- factor(nhanesf$age, labels = c("20-39", "40-59", "60
    -99"))
nhanesf$hyp <- factor(nhanesf$hyp, labels = c("no","yes"))
imp2 <- mice(nhanes, printFlag = FALSE)
fit2 <- with(data = imp2, exp = lm(bmi~hyp+chl))
pool(fit2)
```

```
Call: pool(object = fit2)

Pooled coefficients:
(Intercept)         hyp         chl
     23.613      -1.486       0.027

Fraction of information about the coefficients missing due to
    nonresponse:
(Intercept)         hyp         chl
       0.42        0.25        0.34
```

Compare "real" and 2 imputed sets

```
cbind("F", nhanes,"imp1",complete(imp2,3),"imp2",complete(imp,3))
```

	"F"	age	bmi	hyp	chl	"imp1"	age	bmi	hyp	chl	"imp2"	age	bmi	hyp	chl
1	F	1	NA	NA	NA	imp1	1	35	1	187	imp2	1	30	1	199
2	F	2	23	1	187	imp1	2	23	1	187	imp2	2	23	1	187
3	F	1	NA	1	187	imp1	1	30	1	187	imp2	1	29	1	187
4	F	3	NA	NA	NA	imp1	3	25	1	284	imp2	3	26	1	206
5	F	1	20	1	113	imp1	1	20	1	113	imp2	1	20	1	113
6	F	3	NA	NA	184	imp1	3	20	2	184	imp2	3	26	2	184
7	F	1	22	1	118	imp1	1	22	1	118	imp2	1	22	1	118
8	F	1	30	1	187	imp1	1	30	1	187	imp2	1	30	1	187
9	F	2	22	1	238	imp1	2	22	1	238	imp2	2	22	1	238
10	F	2	NA	NA	NA	imp1	2	27	1	229	imp2	2	28	2	204
11	F	1	NA	NA	NA	imp1	1	30	1	131	imp2	1	33	1	206
12	F	2	NA	NA	NA	imp1	2	30	1	206	imp2	2	22	1	238
13	F	3	22	1	206	imp1	3	22	1	206	imp2	3	22	1	206
14	F	2	29	2	204	imp1	2	29	2	204	imp2	2	29	2	204
15	F	1	30	1	NA	imp1	1	30	1	238	imp2	1	30	1	199
16	F	1	NA	NA	NA	imp1	1	30	1	238	imp2	1	30	1	187
17	F	3	27	2	284	imp1	3	27	2	284	imp2	3	27	2	284
18	F	2	26	2	199	imp1	2	26	2	199	imp2	2	26	2	199
19	F	1	35	1	218	imp1	1	35	1	218	imp2	1	35	1	218
20	F	3	26	2	NA	imp1	3	26	2	218	imp2	3	26	2	206
21	F	1	NA	NA	NA	imp1	1	28	1	238	imp2	1	22	1	118
22	F	1	33	1	229	imp1	1	33	1	229	imp2	1	33	1	229

Compare "real" and 2 imputed sets ...

23	F	1	28	1	131	imp1	1	28	1	131	imp2	1	28	1	131
24	F	3	25	1	NA	imp1	3	25	1	284	imp2	3	25	1	206
25	F	2	27	1	186	imp1	2	27	1	186	imp2	2	27	1	186

How About the Titanic Data?

```
load("/home/pauljohn/SVN/SVN-guides/Rcourse/DataSets/titanic0.rda")
set.seed(234234)
titanic <- titanic0
titanic$sex[sample(nrow(titanic), 400)] <- NA
titanic$pclass[sample(nrow(titanic), 400)] <- NA
titanic$survived[sample(nrow(titanic), 400)] <- NA
miceTitanic <- mice( subset( titanic, select = c('survived', '
    pclass', 'sex', 'age', 'embarked')), m = 10, maxit  = 10,
    printFlag=FALSE)
miceFitTitanic <- with(data = miceTitanic, exp = glm(survived ~
    pclass + sex + age, family = binomial))
pool(miceFitTitanic)
```

```
Call: pool(object = miceFitTitanic)

Pooled coefficients:
(Intercept)      pclass2      pclass3         sex2          age
       3.97        -1.24        -2.81        -2.31        -0.05

Fraction of information about the coefficients missing due to
    nonresponse:
(Intercept)      pclass2      pclass3         sex2          age
       0.92         0.75         0.77         0.81         0.93
```

Here's the error you see if you forget to subset the variables with select

How About the Titanic Data? ...

```
Error: chunk 7 (label=mice70)
Error in nnet.default(X, Y, w, mask = mask, size = 0, skip = TRUE,
    softmax = TRUE, : too many (3210)
weights Execution halted
```

summary of pooled fit

```
round(summary(pool(miceFitTitanic)), 2)
```

	est	se	t	df	Pr(>|t|)	lo 95	hi 95	nmis	fmi	lambda
(Intercept)	3.97	1.03	3.9	10.1	0.00	1.7	6.3	NA	0.92	0.90
pclass2	−1.24	0.40	−3.1	16.7	0.01	−2.1	−0.4	NA	0.75	0.72
pclass3	−2.81	0.41	−6.9	15.8	0.00	−3.7	−1.9	NA	0.77	0.74
sex2	−2.31	0.35	−6.5	13.8	0.00	−3.1	−1.6	NA	0.81	0.79
age	−0.05	0.02	−2.5	9.8	0.03	−0.1	0.0	680	0.93	0.92

The mi Package (Gelman's Columbia U Team)

Yu-Sung Su, Andrew Gelman, Jennifer Hill, Masanao Yajima. 2011. "Multiple Imputation with Diagnostics (mi) in R: Opening Windows into the Black Box". *Journal of Statistical Software.* 45(2)
Kobi Abayomi, Andrew Gelman and Marc Levy. (2008). "Diagnostics for multivariate imputations". *Applied Statistics* 57, Part 3: 273-291.

> *"Generate a multiply imputed matrix applying the elementary functions iteratively to the variables with missingness in the data randomly imputing each variable and looping through until approximate convergence."*

Rough Sketch

- Strategy similar to mice and aregImpute: proceed one-variable-at-a-time
- Predict each variable from each of the others
  - Start with median/mode for NAs
  - Conduct "n.iter" iterations, or until convergence
  - Provides a large set of mi.XXX functions to impute variables of different types
- Draw bootstrap sample to create imputed values
- Allows limited "preprocessing" of numeric variables (similar idea to aregImpute's use of avas)
- As "Opening Windows into the Black Box" implies, this is intended to be less "mysterious," more "informative", and easier to diagnose MI processes.

Type-Dependent Imputation Methods

type	mi name	model
binary	mi.binary	logistic
unordered	mi.categorical	multinomial
ordinal	mi.polr	continuation logistic
continuous	mi.continous	regression
count	mi.count	Bayesian Poisson (w overdispersion)

Interface

```
mi(object, info, n.imp = 3, n.iter = 30, R.hat =
    1.1, max.minutes = 20,
rand.imp.method="bootstrap", run.past.convergence =
    FALSE, seed = NA, check.coef.convergence =
    FALSE, add.noise = noise.control())
```

Steps to Use mi

1. Create an "information table"
2. Use mi to create imputations
   - Runs 1 separate "chain" for each desired imputation
3. pooling methods implemented for common R estimators like "lm.mi","glm.mi","lmer.mi"
   - These cycle through all imputed data frames
   - create estimates for each
4. display or other extractor methods can present results

Information Table for nhanes data with factor variables in it

```
inf <- mi.info(nhanes2)
inf
```

```
  names include order number.mis all.mis
                          type collinear
1   age      Yes    NA          0      No
   unordered-categorical           No
2   bmi      Yes     1          9      No
   positive-continuous          No
3   hyp      Yes     2          8      No
                    binary         No
4   chl      Yes     3         10      No
   positive-continuous          No
```

Can customize variable types at this stage

Run mi

Caution: 2013-05-21 runtime errors were observed with n.imp = 10, only 8 will succeed.

```
miImputeNhanes2 <- mi(nhanes2, info = inf , n.imp = 8,  n.iter =
    500, add.noise = FALSE)
```

Regression: mi Summary for 10 Imputations

```
M1 <- lm.mi(bmi ~ hyp + chl, miImputeNhanes2)
display(M1)
```

```
============================================================
Separate Estimates for each Imputation
============================================================

** Chain 1 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 21.17     4.42
hypyes       0.09     2.64
chl          0.03     0.02
---
n = 25, k = 3
residual sd = 4.72, R-Squared = 0.07

** Chain 2 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 18.71     3.84
hypyes      -1.16     3.15
chl          0.05     0.02
---
n = 25, k = 3
```

Regression: mi Summary for 10 Imputations ...

```
residual sd = 5.18, R-Squared = 0.24

** Chain 3 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 21.79     4.41
hypyes      -2.43     2.54
chl          0.03     0.02
---
n = 25, k = 3
residual sd = 4.79, R-Squared = 0.08

** Chain 4 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 19.37     3.88
hypyes      -0.05     2.41
chl          0.03     0.02
---
n = 25, k = 3
residual sd = 4.26, R-Squared = 0.12

** Chain 5 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 23.78     4.09
```

Regression: mi Summary for 10 Imputations ...

```
hypyes        -0.70      2.78
chl            0.02      0.02
---
n = 25, k = 3
residual sd = 4.40, R-Squared = 0.02

** Chain 6 **
lm(formula = formula, data = mi.data[[i]])
             coef.est coef.se
(Intercept) 16.29     4.66
hypyes      -4.31     2.41
chl          0.06     0.02
---
n = 25, k = 3
residual sd = 4.96, R-Squared = 0.24

** Chain 7 **
lm(formula = formula, data = mi.data[[i]])
             coef.est coef.se
(Intercept) 26.90     3.74
hypyes       1.38     2.05
chl          0.00     0.02
---
n = 25, k = 3
residual sd = 4.09, R-Squared = 0.02
```

Regression: mi Summary for 10 Imputations ...

```
** Chain 8 **
lm(formula = formula, data = mi.data[[i]])
            coef.est coef.se
(Intercept) 28.36     4.70
hypyes      -0.10     3.78
chl         -0.01     0.03
---
n = 25, k = 3
residual sd = 6.24, R-Squared = 0.00

================================================
Pooled Estimates
================================================
lm.mi(formula = bmi ~ hyp + chl, mi.object = miImputeNhanes2)
            coef.est coef.se
(Intercept) 22.04     6.08
hypyes      -0.91     3.33
chl          0.02     0.03
---
```

```
cbind(b=coef(M1), se=se.coef(M1), t=coef(M1)/se.coef(M1))
```

Regression: mi Summary for 10 Imputations ...

```
                  b     se      t
(Intercept) 22.045 6.082  3.62
hypyes      -0.910 3.334 -0.27
chl          0.025 0.033  0.76
```

Titanic, Featuring Kate Winslet and Leonardo DiCaprio

```
load("/home/pauljohn/SVN/SVN-guides/Rcourse/DataSets/titanic0.rda")
set.seed(234234)
titanic <- titanic0
titanic$sex[sample(nrow(titanic), 400)] <- NA
titanic$pclass[sample(nrow(titanic), 400)] <- NA
titanic$survived[sample(nrow(titanic), 400)] <- NA
```

Grab Subset, then compute min.info (prepare for imputation)

Subset required to avoid use of extraneous variables by imputer.
Can customize inf to change variable types, if desired.
Note, mi did not converge with "embarked" included as predictor, so it was omitted here

```
ss <- subset( titanic ,    select = c('survived', 'pclass', 'sex', '
    age'))
inf <- mi.info(ss)
inf
```

```
     names include order number.mis all.mis                    type
         collinear
1 survived     Yes     1        400      No                  binary
           No
2   pclass     Yes     2        400      No unordered-categorical
           No
3      sex     Yes     3        400      No                  binary
           No
4      age     Yes     4        680      No   positive-continuous
           No
```

Create 10 Titanics

```
miImpTitanic <- mi(ss, info = inf, n.imp = 10,  n.iter = 400,
     add.noise = FALSE)
```

n.iter set higher, convergence can take more than 100 iterations

Use "glm.mi" from mi on the List of Imputed Datasets

```
M2 <- glm.mi( survived ~ pclass + sex + age, milmpTitanic, family =
      binomial(link = "logit"))
display(M2)
```

```
============================================
Separate Estimates for each Imputation
============================================

** Chain 1 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   3.70     0.32
pclass2nd    -0.64     0.21
pclass3rd    -4.12     0.24
sexmale      -1.61     0.17
age          -0.04     0.01
---
  n = 1313, k = 5
  residual deviance = 1053.3, null deviance = 1740.9 (difference =
      687.6)

** Chain 2 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   4.73     0.37
```

Use "glm.mi" from mi on the List of Imputed Datasets ...

```
pclass2nd     -1.34       0.23
pclass3rd     -4.03       0.27
sexmale       -2.93       0.20
age           -0.07       0.01
---
  n = 1313, k = 5
  residual deviance = 929.3, null deviance = 1559.7 (difference =
      630.4)

** Chain 3 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   4.85       0.35
pclass2nd     -1.33       0.22
pclass3rd     -2.86       0.23
sexmale       -2.62       0.17
age           -0.09       0.01
---
  n = 1313, k = 5
  residual deviance = 1067.9, null deviance = 1628.9 (difference =
      561.0)

** Chain 4 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   5.30       0.38
```

Use "glm.mi" from mi on the List of Imputed Datasets ...

```
pclass2nd     -1.67      0.24
pclass3rd     -3.39      0.26
sexmale       -3.56      0.20
age           -0.06      0.01
---
  n = 1313, k = 5
  residual deviance = 967.6, null deviance = 1694.5 (difference =
      726.9)

** Chain 5 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   5.52      0.39
pclass2nd     -2.12      0.23
pclass3rd     -4.84      0.28
sexmale       -1.73      0.17
age           -0.08      0.01
---
  n = 1313, k = 5
  residual deviance = 1027.7, null deviance = 1698.3 (difference =
      670.6)

** Chain 6 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   5.43      0.39
```

Use "glm.mi" from mi on the List of Imputed Datasets ...

```
pclass2nd     -1.40      0.24
pclass3rd     -3.46      0.26
sexmale       -3.74      0.21
age           -0.07      0.01
---
  n = 1313, k = 5
  residual deviance = 937.2, null deviance = 1699.5 (difference =
      762.3)

** Chain 7 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   4.55      0.34
pclass2nd     -1.58      0.21
pclass3rd     -4.32      0.26
sexmale       -1.74      0.16
age           -0.06      0.01
---
  n = 1313, k = 5
  residual deviance = 1080.9, null deviance = 1678.8 (difference =
      597.9)

** Chain 8 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   4.30      0.32
```

Use "glm.mi" from mi on the List of Imputed Datasets ...

```
pclass2nd     -0.88      0.21
pclass3rd     -3.51      0.22
sexmale       -2.30      0.17
age           -0.04      0.01
---
  n = 1313, k = 5
  residual deviance = 1185.1, null deviance = 1796.8 (difference =
      611.8)

** Chain 9 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   5.00      0.38
pclass2nd     -1.54      0.25
pclass3rd     -3.26      0.26
sexmale       -3.90      0.21
age           -0.05      0.01
---
  n = 1313, k = 5
  residual deviance = 918.0, null deviance = 1691.9 (difference =
      773.9)

** Chain 10 **
glm(formula = formula, family = family, data = mi.data[[i]])
             coef.est coef.se
(Intercept)   4.74      0.35
```

Use "glm.mi" from mi on the List of Imputed Datasets ...

```
pclass2nd    -1.48      0.22
pclass3rd    -4.09      0.25
sexmale      -1.48      0.16
age          -0.08      0.01
---
  n = 1313, k = 5
  residual deviance = 1083.3, null deviance = 1654.8 (difference =
      571.5)

============================================================
Pooled Estimates
============================================================
glm.mi(formula = survived ~ pclass + sex + age, mi.object =
    milmpTitanic,
    family = binomial(link = "logit"))
             coef.est coef.se
(Intercept)   4.81     0.68
pclass2nd    -1.40     0.48
pclass3rd    -3.79     0.67
sexmale      -2.56     1.00
age          -0.06     0.02
---
```

```
(miTitanicResult <- cbind(b = coef(M2), se = se.coef(M2), t = coef(
    M2)/se.coef(M2)))
```

```
                b    se    t
(Intercept)  4.812 0.681  7.1
pclass2nd   -1.399 0.483 -2.9
pclass3rd   -3.789 0.668 -5.7
sexmale     -2.560 0.996 -2.6
age         -0.064 0.019 -3.3
```

The Hmisc & rms Packages

Harrell, Frank E. 2010. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer.
Frank E Harrell Jr (2013). rms: Regression Modeling Strategies. R package version 3.6-3.

> *'aregImpute' takes all aspects of uncertainty in the imputations into account by using the bootstrap to approximate the process of drawing predicted values from a full Bayesian predictive distribution. Different bootstrap resamples are used for each of the multiple imputations, i.e., for the 'i'th imputation of a sometimes missing variable, 'i=1,2,... n.impute', a flexible additive model is fitted on a sample with replacement from the original data and this model is used to predict all of the original missing and non-missing values for the target variable."*

Rough Sketch

Predict each variable from each of the others

- Start with random selections for NAs

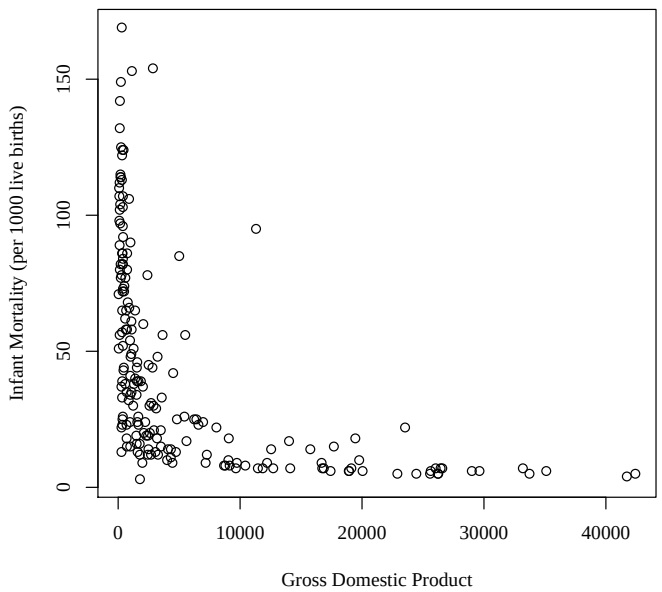
Do this "burnin"+"n.impute" times

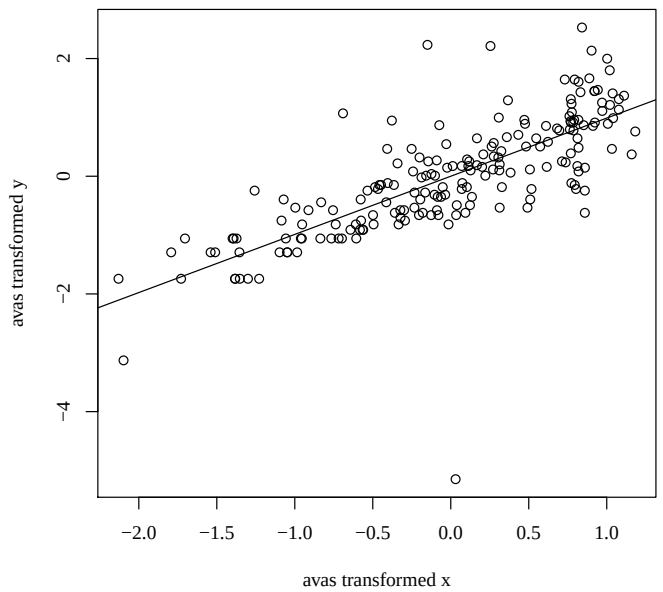
- Draw bootstrap sample, fit a "flexible" model to it, predict outcomes for all cases from that model.
- Default uses predictive mean matching to select an imputed value for each NA. (regression extrapolation is alternative)

Note: Special emphasis on nonlinearity in imputation fitting (similar to avas)

What's Avas?

From:Squeeze and Stretch Variables	To: Estimate, Convert Back





Interface

```
areglmpute(formula, data, subset, n.impute = 5,
    group = NULL, nk = 3, tlinear = TRUE, type = c(
    'pmm', 'regression'), match = c('weighted', '
    closest'), fweighted = 0.2, curtail = TRUE,
    boot.method = c('simple', 'approximate bayesian
    '), burnin = 3, x = FALSE, pr = TRUE, plotTrans
     = FALSE, tolerance = NULL, B = 75)
```

Important Detail about "rms"

- Prof. Harrell is a long-standing, highly distinguished programmer and statistician (SAS PROC Logistic in mid 1980s)
- He has developed his own set of fitting functions in S over 2 decades and they are not perfectly interchangeable with R functions
  - rms "ols" is not exactly same as R's "lm"
  - rms "lrm" is not exactly same as R's glm(y~x, family=binomial(link="logit"))
  - aregImpute and other rms functions are tailored for rms "fitting routines", but tolerate some R routines (with warnings).
- Summary and Plotting functions for "rms" objects are usually expecting different options than functions in base R

Syntax Example

```
[1] "f <- areglmpute(~ age + bmi + hyp + chl, data=
    nhanesf, nk=0)"
```

- Can't actually run that, though, because of this error (which I have not solved)

```
Error in areglmpute(~age + bmi + hyp + chl, data =
    nhanesf, nk = 0) : a bootstrap resample had too
     few unique values of the following variables:
    age Execution halted
```

```
Warning in areglmpute(~age + bmi + hyp + chl, data
    = nhanesf, nk = 1) : hyp has the following
    levels with < 5 observations: yes Consider
    using the group parameter to balance bootstrap
    samples
```

If that did work, we would run fit.mult.impute

```
fmi <- fit.mult.impute(bmi ~ hyp + chl, lm, f, data
    = nhanesf)
```

```
sqrt(diag(vcov(fmi)))
```

- Even if aregImpute did create imputations, it would be accompanied by this warning

```
Warning in fit.mult.impute(bmi ~ hyp + chl, lm, f,
    data = nhanesf) : Not using a Design fitting
    function; summary(fit) will use standard errors
    , t, P from last imputation only. Use vcov(fit)
     to get the correct covariance matrix, sqrt(
    diag(vcov(fit))) to get s.e.
```

- Caused by my use of R's "lm", rather than rms's own function "ols"

The Titanic Rises Again

```
load("/home/pauljohn/SVN/SVN-guides/Rcourse/DataSets/titanic0.rda")
set.seed(234234)
titanic <- titanic0
titanic$sex[sample(nrow(titanic), 400)] <- NA
titanic$pclass[sample(nrow(titanic), 400)] <- NA
titanic$survived[sample(nrow(titanic), 400)] <- NA
rmsImputeTitanic <- aregImpute(~ survived + pclass + sex + age +
     embarked, n.impute=10, data=titanic, nk=3)
```

```
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Iteration 9
Iteration 10
Iteration 11
Iteration 12
Iteration 13
```

Use glm in fit.mult.impute

```
rmsFmiTitanic <- fit.mult.impute( survived ~ pclass + sex + age,
    glm, family=binomial(link=logit), rmsImputeTitanic, data=
    titanic, fit.reps=TRUE)
```

```
Variance Inflation Factors Due to Imputation:

(Intercept)    pclass2nd    pclass3rd      sexmale          age
        2.8          2.1          4.5          1.9          2.5

Rate of Missing Information:

(Intercept)    pclass2nd    pclass3rd      sexmale          age
       0.65         0.53         0.78         0.48         0.59

d.f. for t-distribution for Tests of Single Coefficients:

(Intercept)    pclass2nd    pclass3rd      sexmale          age
         22           32           15           39           26

The following fit components were averaged over the 10 model fits:

  fitted.values linear.predictors
```

```
summary(rmsFmiTitanic)
```

Use glm in fit.mult.impute ...

```
Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.143  -0.735  -0.386   0.680   2.901

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.24518    0.30409   13.96  < 2e-16 ***
pclass2nd   -1.37042    0.20975   -6.53  6.4e-11 ***
pclass3rd   -2.85903    0.19864  -14.39  < 2e-16 ***
sexmale     -2.37998    0.15689  -15.17  < 2e-16 ***
age         -0.05526    0.00569   -9.71  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1680.1  on 1312  degrees of freedom
Residual deviance: 1148.6  on 1308  degrees of freedom
AIC: 1159

Number of Fisher Scoring iterations: 5
```

Recall the Cautionary Warning about Fitting Functions?

- fit.mult.impute warns the user that when a fitting routine is not from rms, then the standard errors and significance tests are based only on the last fitted model
- One should instead extract the standard errors from the covariance matrix
- Which I do on the next slide

Create a Homemade Summary for the Fit.Mult.Impute Output

```
sqrt(diag(vcov(rmsFmiTitanic)))
```

```
(Intercept)    pclass2nd    pclass3rd      sexmale          age
     0.3041       0.2097       0.1986       0.1569       0.0057
```

```
(rmsTitanicResult <- cbind(b=rmsFmiTitanic$coefficients, se=diag(
    vcov(rmsFmiTitanic))))
```

```
                  b       se
(Intercept)   4.245 9.2e-02
pclass2nd    -1.370 4.4e-02
pclass3rd    -2.859 3.9e-02
sexmale      -2.380 2.5e-02
age          -0.055 3.2e-05
```

Here's a Problem: I don't believe the se result. Compare mitools

```
require(mitools)
rmsbetas <- MIextract(rmsFmiTitanic$fits, fun=coef)
rmsvars <- MIextract(rmsFmiTitanic$fits, fun=vcov)
rmsTitanicMItools <- summary(MIcombine(rmsbetas,rmsvars))
```

```
Multiple imputation results:
      MIcombine.default(rmsbetas, rmsvars)
             results      se (lower upper) missInfo
(Intercept)   4.245 0.5458  3.112  5.378      68 %
pclass2nd    -1.370 0.3132 -2.008 -0.733      55 %
pclass3rd    -2.859 0.4361 -3.789 -1.929      80 %
sexmale      -2.380 0.2273 -2.840 -1.920      51 %
age          -0.055 0.0093 -0.074 -0.036      62 %
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model

```
for(i in 1:length(rmsFmiTitanic$fits)) print(summary(rmsFmiTitanic$
    fits[[i]]))
```

```
Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.318  -0.722  -0.375   0.624   3.065

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.27977    0.33643   12.72  < 2e-16 ***
pclass2nd    -1.30687    0.21447   -6.09  1.1e-09 ***
pclass3rd    -2.77282    0.20172  -13.75  < 2e-16 ***
sexmale      -2.34844    0.16246  -14.46  < 2e-16 ***
age          -0.05740    0.00612   -9.38  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1680.1  on 1312  degrees of freedom
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
Residual deviance: 1148.6  on 1308  degrees of freedom
AIC: 1159

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.488  -0.657  -0.350   0.629   2.976

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.70439    0.32223   14.60  < 2e-16 ***
pclass2nd   -1.69747    0.21563   -7.87  3.5e-15 ***
pclass3rd   -2.86445    0.20043  -14.29  < 2e-16 ***
sexmale     -2.11856    0.16525  -12.82  < 2e-16 ***
age         -0.06367    0.00564  -11.29  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
    Null deviance: 1698.3  on 1312  degrees of freedom
Residual deviance: 1103.0  on 1308  degrees of freedom
AIC: 1113

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.522  -0.584  -0.326   0.549   2.669

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.15318    0.36868   13.98  < 2e-16 ***
pclass2nd   -1.68279    0.23263   -7.23  4.7e-13 ***
pclass3rd   -3.72679    0.23826  -15.64  < 2e-16 ***
sexmale     -2.52675    0.17290  -14.61  < 2e-16 ***
age         -0.06950    0.00684  -10.16  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1691.9  on 1312  degrees of freedom
Residual deviance: 1050.4  on 1308  degrees of freedom
AIC: 1060

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.573  -0.648  -0.351   0.585   2.765

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.26266    0.31396   13.58  < 2e-16 ***
pclass2nd   -1.43351    0.21260   -6.74  1.6e-11 ***
pclass3rd   -3.00432    0.21207  -14.17  < 2e-16 ***
sexmale     -2.47417    0.16366  -15.12  < 2e-16 ***
age         -0.05497    0.00572   -9.61  < 2e-16 ***
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1689.4  on 1312  degrees of freedom
Residual deviance: 1108.5  on 1308  degrees of freedom
AIC: 1119

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.523  -0.672  -0.375   0.578   2.588

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.08354    0.31901   12.80   <2e-16 ***
pclass2nd   -1.46783    0.21463   -6.84    8e-12 ***
pclass3rd   -2.91258    0.20744  -14.04   <2e-16 ***
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
sexmale      −2.50116    0.16286  −15.36   <2e−16 ***
age          −0.04957    0.00576   −8.61   <2e−16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1678.8  on 1312  degrees of freedom
Residual deviance: 1119.7  on 1308  degrees of freedom
AIC: 1130

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
−2.637  −0.750  −0.404   0.616   2.819

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.16399    0.33749   12.34  < 2e−16 ***
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
pclass2nd    -1.38169    0.21875   -6.32  2.7e-10 ***
pclass3rd    -2.82804    0.20494  -13.80  < 2e-16 ***
sexmale      -2.30791    0.16029  -14.40  < 2e-16 ***
age          -0.05521    0.00624   -8.85  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1686.8  on 1312  degrees of freedom
Residual deviance: 1172.5  on 1308  degrees of freedom
AIC: 1182

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.810  -0.747  -0.186   0.629   2.685

Coefficients:
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.0348     0.2995   13.47  < 2e-16 ***
pclass2nd    -1.2541     0.2085   -6.01  1.8e-09 ***
pclass3rd    -2.4904     0.1952  -12.76  < 2e-16 ***
sexmale      -2.2320     0.1633  -13.66  < 2e-16 ***
age          -0.0525     0.0056   -9.37  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1694.5  on 1312  degrees of freedom
Residual deviance: 1100.1  on 1308  degrees of freedom
AIC: 1110

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.497  -0.678  -0.323   0.529   2.812
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.23938    0.34465   12.30  < 2e-16 ***
pclass2nd   -1.28286    0.22106   -5.80  6.5e-09 ***
pclass3rd   -2.95459    0.21243  -13.91  < 2e-16 ***
sexmale     -2.62029    0.16949  -15.46  < 2e-16 ***
age         -0.05305    0.00634   -8.36  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1669.2  on 1312  degrees of freedom
Residual deviance: 1075.9  on 1308  degrees of freedom
AIC: 1086

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)

Deviance Residuals:
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
   Min      1Q  Median      3Q     Max
-2.417  -0.693  -0.330   0.607   2.893

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.75758    0.29209   12.86  < 2e-16 ***
pclass2nd   -0.97306    0.20516   -4.74  2.1e-06 ***
pclass3rd   -2.36932    0.19073  -12.42  < 2e-16 ***
sexmale     -2.34050    0.15930  -14.69  < 2e-16 ***
age         -0.04948    0.00538   -9.19  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1680.1  on 1312  degrees of freedom
Residual deviance: 1133.4  on 1308  degrees of freedom
AIC: 1143

Number of Fisher Scoring iterations: 5


Call:
fitter(formula = formula, family = ..1, data = completed.data)
```

Notice the "fit.reps=T" option? It allows inspection of each fitted model ...

```
Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.458  -0.696  -0.376   0.618   2.838

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.77253    0.30409   12.41  < 2e-16 ***
pclass2nd   -1.22397    0.20975   -5.84  5.4e-09 ***
pclass3rd   -2.66704    0.19864  -13.43  < 2e-16 ***
sexmale     -2.33004    0.15689  -14.85  < 2e-16 ***
age         -0.04719    0.00569   -8.30  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1680.1  on 1312  degrees of freedom
Residual deviance: 1148.6  on 1308  degrees of freedom
AIC: 1159

Number of Fisher Scoring iterations: 5
```

Using "lrm" from rms package (instead of glm from R)

```
fmi2 <- fit.mult.impute( survived ~ pclass + sex + age, lrm,
    rmsImputeTitanic, data= titanic, fit.reps=TRUE)
```

```
Variance Inflation Factors Due to Imputation:

 Intercept pclass=2nd pclass=3rd   sex=male         age
       2.8         2.1         4.5         1.9         2.5

Rate of Missing Information:

 Intercept pclass=2nd pclass=3rd   sex=male         age
      0.65        0.53        0.78        0.48        0.59

d.f. for t-distribution for Tests of Single Coefficients:

 Intercept pclass=2nd pclass=3rd   sex=male         age
        22          32          15          39          26

The following fit components were averaged over the 10 model fits:

  stats linear.predictors
```

```
fmi2
```

Using "lrm" from rms package (instead of glm from R) ...

```
Logistic Regression Model

fitter(formula = formula, data = completed.data)

                      Model Likelihood     Discrimination    Rank
                          Discrim.
                             Ratio Test          Indexes
                                Indexes
Obs            1313    LR chi2     568.84    R2        0.486    C
          0.862
 0              869    d.f.             4    g         2.123    Dxy
         0.724
 1              444    Pr(> chi2) <0.0001    gr        8.441    gamma
        0.726
max |deriv| 9e-07                            gp        0.328    tau-a
       0.326
                                             Brier     0.134

             Coef    S.E.   Wald Z Pr(>|Z|)
Intercept     4.2452 0.5458   7.78 <0.0001
pclass=2nd   -1.3704 0.3132  -4.38 <0.0001
pclass=3rd   -2.8590 0.4361  -6.56 <0.0001
sex=male     -2.3800 0.2273 -10.47 <0.0001
age          -0.0553 0.0093  -5.93 <0.0001
```

Using "lrm" from rms package (instead of glm from R) ...

- Please note: the standard errors in the output based on lrm match the std.errors estimated by MItools. Thus I conclude sqrt(diag(cov(fit.mult.impute.object) did not give correct results.

First Try at summary fails:

```
summary(fmi2)
```

```
Error in summary.rms(fmi2) : adjustment values not
    defined here or with datadist for pclass sex
    age Execution halted Error: Cannot convert file
```

rms fitters require a "datadist" object must be defined

```
d <- datadist(titanic)
options(datadist="d")
summary(fmi2)
```

Effects Response : survived

Factor	Low	High	Diff.	Effect	S.E.	Lower 0.95	Upper 0.95
age	21	41	20	−1.11	0.19	−1.47	−0.74
Odds Ratio	21	41	20	0.33	NA	0.23	0.48
pclass − 1st:3rd	3	1	NA	2.86	0.44	2.00	3.71
Odds Ratio	3	1	NA	17.44	NA	7.42	41.01
pclass − 2nd:3rd	3	2	NA	1.49	0.32	0.87	2.11
Odds Ratio	3	2	NA	4.43	NA	2.38	8.26
sex − female:male	2	1	NA	2.38	0.23	1.93	2.83
Odds Ratio	2	1	NA	10.80	NA	6.92	16.87

What Should You Do Now?

- Can't ignore "missing data problem" any more, but
- No "lead pipe cinch" solution exists at this time
- I wish there were decisive results comparing these algorithms to find out "which one is best."
- Until then, I expect researchers will use whatever tools are prevalent in their fields

Encouraging Titanic News: 4 Results are Mostly the Same

Amelia, areglmpute

mi , mice

```
cbind("Amelia"="Amelia",df1[ , 1:2 ], "rms"="rms",
      rmsTitanicMItools[,1:2])
```

	Amelia	est	std.err	rms	results	se
(Intercept)	Amelia	4.03	0.532	rms	4.245	0.5458
pclass2nd	Amelia	−1.44	0.258	rms	−1.370	0.3132
pclass3rd	Amelia	−2.71	0.308	rms	−2.859	0.4361
sexmale	Amelia	−2.53	0.175	rms	−2.380	0.2273
age	Amelia	−0.05	0.011	rms	−0.055	0.0093

```
cbind( "mi"="mi", round(miTitanicResult,2), "mice"="mice", round(
      summary(pool(miceFitTitanic)),2)[,1:3])
```

	mi	b	se	t	mice	est	se	t
(Intercept)	"mi"	"4.81"	"0.68"	"7.07"	"mice"	"3.97"	"1.03"	"3.87"
pclass2nd	"mi"	"−1.4"	"0.48"	"−2.89"	"mice"	"−1.24"	"0.4"	"−3.12 "
pclass3rd	"mi"	"−3.79"	"0.67"	"−5.67"	"mice"	"−2.81"	"0.41"	"−6.92 "
sexmale	"mi"	"−2.56"	"1"	"−2.57"	"mice"	"−2.31"	"0.35"	"−6.55 "
age	"mi"	"−0.06"	"0.02"	"−3.31"	"mice"	"−0.05"	"0.02"	"−2.48 "