Using Multiple Hot Deck Data Sets for Inference

Skyler Cranmer
Ohio State University

Jeff Gill
Washington University St. Louis

Natalie Jackson
The Huffington Post

Andreas Murr
University of Oxford

David A. Armstrong II
University of Wisconsin-Milwaukee

November 19, 2015

This document will walk you through some of the methods you could use to generate pooled model results that account for both sampling variability and across imputation variability. The package `hot.deck` does not come with a set of functions to do inference, so we will show you how you could use the data generated by `hot.deck` in combination with `glm.mids` (and similarly `lm.mids`) from the `mice` package, `zelig` from the `Zelig` package and by using `MIcombine` from the `mitools` package on a list of model objects.

1 Generating Imputations

The data we will use come from Poe, Tate and Keith (1999) dealing with democracy and state repression. First we need to call the `hot.deck` routine on the dataset.

```
> library(hot.deck)
> data(isq99)
> out <- hot.deck(isq99, sdCutoff=3, IDvars = c("IDORIGIN", "YEAR"))
```

This shows us that there are still 47 observations with fewer than 5 donors. Using a different method or further widening the `sdCutoff` parameter may alleviate the problem. If you want to see the frequency distribution of the number of donors, you could look at:

```
> numdonors <- sapply(out$donors, length)
> numdonors <- sapply(out$donors, length)
> numdonors <- ifelse(numdonors > 5, 6, numdonors)
> numdonors <- factor(numdonors, levels=1:6, labels=c(1:5, ">5"))
> table(numdonors)
```

```
numdonors
   1    2    3    4    5   >5
  18   10   11    6   20 4596
```

Before running a model, three variables have to be created from those existing. Generally, if variables are deterministic functions of other variables (e.g., transformations, lags, etc...) it is advisable to impute the constituent variables of the calculations and then do the calculations after the fact. Here, we need to lag the `AI` variable and create percentage change variables for both population and per-capita GNP. First, to create the lag of `AI`, `PCGNP` and `LPOP`. To do this, we will make a little function.

```
> tscslag <- function(dat, x, id, time){
+         obs <- apply(dat[, c(id, time)], 1, paste, collapse=".")
+         tm1 <- dat[[time]] - 1
+         lagobs <- apply(cbind(dat[[id]], tm1), 1, paste, collapse=".")
+         lagx <- dat[match(lagobs, obs), x]
+ }
> for(i in 1:length(out$data)){
+     out$data[[i]]$lagAI <- tscslag(out$data[[i]], "AI", "IDORIGIN", "YEAR")
+     out$data[[i]]$lagPCGNP <- tscslag(out$data[[i]], "PCGNP", "IDORIGIN", "YEAR")
+     out$data[[i]]$lagLPOP <- tscslag(out$data[[i]], "LPOP", "IDORIGIN", "YEAR")
+ }
```

Now, we can use the lagged values of `PCGNP` and `LPOP`, to create percentage change variables:

```
> for(i in 1:length(out$data)){
+     out$data[[i]]$pctchgPCGNP <- with(out$data[[i]], c(PCGNP-lagPCGNP)/lagPCGNP)
+     out$data[[i]]$pctchgLPOP <- with(out$data[[i]], c(LPOP-lagLPOP)/lagLPOP)
+ }
```

2 Running Models on Multiple Hot Decking Result

2.1 Using Zelig

In version ≥ 5.0 of `Zelig`, the output from `hot.deck` will have to be converted into a format that looks like Amelia's. You can do this as follows:

```
> out <- hd2amelia(out)
```

Then, with the output in the appropriate format, we can use `Zelig` to do the modeling.

```
> library(Zelig)
> z <- zelig(AI ~ lagAI + pctchgPCGNP + PCGNP + pctchgLPOP + LPOP + MIL2 + LEFT +
+     BRIT + POLRT + CWARCOW + IWARCOW2, data=out, model="normal", cite=FALSE)
> summary(z)

Model: Combined Imputations
              Estimate Std.Error  z value  Pr(>|z|)
(Intercept)  5.383e-01 1.313e-01  4.10080 4.117e-05 ***
lagAI        4.557e-01 1.865e-02 24.43109 0.000e+00 ***
pctchgPCGNP  5.863e-03 6.297e-03  0.93102 3.518e-01
PCGNP       -2.117e-05 3.157e-06 -6.70731 1.982e-11 ***
pctchgLPOP   1.147e-01 2.133e+00  0.05377 9.571e-01
LPOP         7.506e-02 8.997e-03  8.34314 0.000e+00 ***
MIL2         1.064e-01 4.820e-02  2.20725 2.730e-02   .
LEFT        -1.407e-01 5.288e-02 -2.66131 7.784e-03   *
BRIT        -1.248e-01 3.700e-02 -3.37393 7.410e-04  **
POLRT       -7.196e-02 9.603e-03 -7.49344 6.706e-14 ***
CWARCOW      6.305e-01 6.052e-02 10.41649 0.000e+00 ***
IWARCOW2     1.930e-01 5.554e-02  3.47513 5.106e-04  **
```

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

For results from individual imputed datasets, use summary(x, subset = i:j)
Next step: Use 'setx' method
```

Note that the summary indicates that the results have been combined across 5 multiply imputed datasets.

2.2 Using MIcombine

You can use the `MIcombine` command from the `mitools` package to generate inferences, too. Here, you have to produce a list of model estimates and the function will combine across the different results.

```
> # initialize list
> results <- list()
> # loop over imputed datasets
> for(i in 1:length(out$imputations)){
+     results[[i]] <- lm(AI ~ lagAI + pctchgPCGNP + PCGNP + pctchgLPOP + LPOP + MIL2 + LEFT +
+     BRIT + POLRT + CWARCOW + IWARCOW2, data=out$imputations[[i]])
+ }
> library(mitools)
> summary(MIcombine(results))

Multiple imputation results:
      MIcombine.default(results)
                  results           se        (lower        upper) missInfo
(Intercept)  5.383423e-01 1.312774e-01  2.808194e-01  7.958653e-01      5 %
lagAI        4.557112e-01 1.865292e-02  4.183012e-01  4.931212e-01     30 %
pctchgPCGNP  5.863086e-03 6.297460e-03 -9.242674e-03  2.096885e-02     83 %
PCGNP       -2.117354e-05 3.156786e-06 -2.746353e-05 -1.488354e-05     25 %
pctchgLPOP   1.147025e-01 2.133266e+00 -5.293916e+00  5.523321e+00     90 %
LPOP         7.506090e-02 8.996719e-03  5.723913e-02  9.288267e-02     20 %
MIL2         1.063827e-01 4.819685e-02  4.205250e-03  2.085602e-01     55 %
LEFT        -1.407167e-01 5.287501e-02 -2.494111e-01 -3.202233e-02     43 %
BRIT        -1.248292e-01 3.699810e-02 -1.998190e-01 -4.983930e-02     36 %
POLRT       -7.196185e-02 9.603319e-03 -9.127034e-02 -5.265337e-02     32 %
CWARCOW      6.304523e-01 6.052444e-02  5.081782e-01  7.527264e-01     35 %
IWARCOW2     1.929974e-01 5.553675e-02  8.321765e-02  3.027771e-01     18 %
```

2.3 Using mids

The final method for combining results is to convert the data object returned by the `hot.deck` function to an object of class `mids`. This can be done with the `datalist2mids` function from the `miceadds` package.

```
> library(miceadds)
> out.mids <- datalist2mids(out$imputations)

----
.....

> s <- summary(pool(lm.mids(AI ~ lagAI + pctchgPCGNP + PCGNP + pctchgLPOP + LPOP + MIL2 + LEFT +
+ BRIT + POLRT + CWARCOW + IWARCOW2, data=out.mids)))
> round(s, 4)
```

	est	se	t	df	Pr(>|t|)	lo 95	hi 95	nmis	fmi	lambda
(Intercept)	0.5349	0.1375	3.8898	193.1376	0.0001	0.2637	0.8061	NA	0.1472	0.1384
lagAI	0.4584	0.0186	24.6999	57.4536	0.0000	0.4213	0.4956	179	0.2849	0.2604
pctchgPCGNP	0.0031	0.0055	0.5540	4.9469	0.6037	-0.0111	0.0172	179	0.9194	0.8923
PCGNP	0.0000	0.0000	-6.0321	24.4393	0.0000	0.0000	0.0000	391	0.4454	0.4018
pctchgLPOP	-0.0422	1.4431	-0.0292	12.1091	0.9772	-3.1833	3.0990	179	0.6287	0.5720
LPOP	0.0750	0.0094	8.0185	61.2519	0.0000	0.0563	0.0937	63	0.2753	0.2520
MIL2	0.0938	0.0483	1.9418	15.8202	0.0702	-0.0087	0.1962	265	0.5533	0.5002
LEFT	-0.1370	0.0528	-2.5930	25.8741	0.0154	-0.2456	-0.0284	212	0.4326	0.3904
BRIT	-0.1306	0.0357	-3.6541	49.0744	0.0006	-0.2024	-0.0588	208	0.3098	0.2822
POLRT	-0.0716	0.0103	-6.9479	27.2527	0.0000	-0.0927	-0.0504	329	0.4212	0.3803
CWARCOW	0.6180	0.0621	9.9470	32.9903	0.0000	0.4916	0.7444	129	0.3816	0.3452
IWARCOW2	0.1914	0.0531	3.6062	447.5809	0.0003	0.0871	0.2957	146	0.0905	0.0864

References

Poe, Steven, C. Neal Tate and Linda Camp Keith. 1999. "Repression of the Human Right to Personal Integrity Revisited: A Global, Cross-National Study Covering the Years 1976-1993." *International Studies Quarterly* 43:291–313.