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

September 15, 2014

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

Now that we have an object of class `mi` and `list`, Zelig can use those data to estimate a model:

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

  Model: normal
  Number of multiply imputed data sets: 5

Combined results:

Call:
glm(formula = formula, weights = weights, family = gaussian,
    model = F, data = data)

Coefficients:
                    Value   Std. Error     t-stat      p-value
(Intercept)  5.064970e-01 1.443369e-01  3.5091292 7.772748e-04
lagAI        4.765268e-01 1.862760e-02 25.5817521 2.221156e-31
pctchgPCGNP  8.136076e-03 4.445418e-03  1.8302164 8.751409e-02
PCGNP       -2.022343e-05 3.373759e-06 -5.9943303 8.569741e-07
pctchgLPOP  -6.795762e-01 1.141224e+00 -0.5954803 5.644959e-01
LPOP         7.218116e-02 9.393669e-03  7.6840219 2.941495e-10
MIL2         1.030845e-01 6.294269e-02  1.6377520 1.405694e-01
LEFT        -1.177515e-01 4.747111e-02 -2.4804880 1.591020e-02
BRIT        -1.202226e-01 3.576720e-02 -3.3612519 1.570913e-03
```

```
POLRT       -6.639945e-02 1.240496e-02 -5.3526517 1.738451e-04
CWARCOW      5.995567e-01 5.952989e-02 10.0715243 2.943470e-13
IWARCOW2     1.693510e-01 6.512539e-02  2.6003838 1.522599e-02

For combined results from datasets i to j, use summary(x, subset = i:j).
For separate results, use print(summary(x), subset = i:j).
```

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$data)){
+     results[[i]] <- lm(AI ~ lagAI + pctchgPCGNP + PCGNP + pctchgLPOP + LPOP + MIL2 + LEFT +
+     BRIT + POLRT + CWARCOW + IWARCOW2, data=out$data[[i]])
+ }
> library(mitools)
> summary(MIcombine(results))

Multiple imputation results:
      MIcombine.default(results)
                 results           se        (lower        upper) missInfo
(Intercept)  5.064970e-01 1.443369e-01  2.187925e-01  7.942014e-01     26 %
lagAI        4.765268e-01 1.862760e-02  4.391593e-01  5.138942e-01     30 %
pctchgPCGNP  8.136076e-03 4.445418e-03 -1.353792e-03  1.762594e-02     57 %
PCGNP       -2.022343e-05 3.373759e-06 -2.707827e-05 -1.336858e-05     38 %
pctchgLPOP  -6.795762e-01 1.141224e+00 -3.215521e+00  1.856368e+00     68 %
LPOP         7.218116e-02 9.393669e-03  5.335386e-02  9.100846e-02     30 %
MIL2         1.030845e-01 6.294269e-02 -4.237078e-02  2.485398e-01     76 %
LEFT        -1.177515e-01 4.747111e-02 -2.126861e-01 -2.281699e-02     28 %
BRIT        -1.202226e-01 3.576720e-02 -1.922217e-01 -4.822348e-02     32 %
POLRT       -6.639945e-02 1.240496e-02 -9.343251e-02 -3.936639e-02     63 %
CWARCOW      5.995567e-01 5.952989e-02  4.797569e-01  7.193566e-01     32 %
IWARCOW2     1.693510e-01 6.512539e-02  3.541225e-02  3.032898e-01     44 %
```

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$data)

----
.....

> 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)  0.4545 0.1635  2.7794  24.1462   0.0104  0.1171  0.7918   NA 0.4481 0.4042
lagAI        0.4790 0.0170 28.1376 179.4457   0.0000  0.4454  0.5126  179 0.1533 0.1439
pctchgPCGNP  0.0086 0.0058  1.4824   7.6536   0.1782 -0.0049  0.0222  179 0.7722 0.7196
PCGNP        0.0000 0.0000 -4.7704  13.7471   0.0003  0.0000  0.0000  391 0.5920 0.5367
pctchgLPOP  -0.5017 0.8088 -0.6204  17.0669   0.5432 -2.2076  1.2042  179 0.5331 0.4814
LPOP         0.0752 0.0103  7.2779  23.9339   0.0000  0.0539  0.0965   63 0.4501 0.4060
MIL2         0.0920 0.0655  1.4033   7.3472   0.2014 -0.0615  0.2455  270 0.7858 0.7344
LEFT        -0.1339 0.0489 -2.7384  42.6967   0.0090 -0.2325 -0.0353  200 0.3334 0.3029
BRIT        -0.1100 0.0344 -3.1978  68.2179   0.0021 -0.1787 -0.0414  203 0.2599 0.2385
POLRT       -0.0682 0.0135 -5.0508   9.4816   0.0006 -0.0985 -0.0379  330 0.7032 0.6466
CWARCOW      0.6019 0.0630  9.5597  28.4847   0.0000  0.4730  0.7308  126 0.4118 0.3719
IWARCOW2     0.1723 0.0588  2.9315  57.5494   0.0048  0.0546  0.2900  116 0.2846 0.2602
```

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.