FHDI: An R Package for Fractional Hot Deck Imputation

by Jongho Im, In Ho Cho, and Jae Kwang Kim

Abstract Fractional hot deck imputation (FHDI), proposed by Kalton and Kish (1984) and investigated by Kim and Fuller (2004), is a tool for handling item nonresponse in survey sampling. In FHDI, each missing item is filled with multiple observed values yielding a single completed data set for subsequent analyses. An R package **FHDI** is developed to perform FHDI and also the fully efficient fractional imputation (FEFI) method of (Fuller and Kim, 2005) to impute multivariate missing data with arbitrary missing patterns. FHDI substitutes missing items with a few observed values jointly obtained from a set of donors whereas the FEFI uses all the possible donors. This paper introduces **FHDI** as a tool for implementing the multivariate version of fractional hot deck imputation discussed in Im et al. (2015) as well as FEFI. For variance estimation of FHDI and FEFI, the Jackknife method is implemented, and replicated weights are provided as a part of the output.

Introduction

Incomplete data are common in survey sampling, biomedical, and social sciences. Naive analysis with only complete cases, conducted by removing all the cases with any missing items, is exposed to nonresponse bias unless the missing data mechanism is missing completely at random (Rubin, 1976). Even if the complete cases can be treated as a complete random sample, the complete case analysis is inefficient, as all the partially observed cases are ignored. To incorporate these partial observations, we may consider an imputation technique in which the missing items are filled in with plausible values.

Imputation is often classified into single imputation and repeated imputation on the basis of the number of imputed values per each missing value. Several values are assigned to each missing value in repeated imputation, while a single value is imputed in single imputation. Although single imputation is often preferred in practice, due to its convenience, it does not necessarily preserve the distribution of the original data. Thus, single imputation is inadequate for general purpose estimation.

There are two popular methods in the repeated imputation: multiple imputation and fractional imputation. Multiple imputation, proposed by Rubin (1987, 1996) produces multiply imputed data sets as imputation output. Many multiple imputation methods are already available in R, for example, **mice** (van Buuren and Groothuis-Oudshoorn, 2011), **mi** (Su et al., 2011), **Amelia** (Honaker et al., 2011), and **VIM** (Kowarik and Templ, 2016). Fractional imputation, initially proposed by Kalton and Kish (1984) and extensively discussed in Fay (1996), Kim and Fuller (2004), Durrant and Skinner (2006), and Kim (2011), creates a single completed data set with fractional weights after imputation. The size of the imputed data is always larger than that of the incomplete input data. However, there is no suitable R package for implementing fractional imputation. So far, the only publicly available software for doing fractional imputation is the SAS procedure `SURVEYIMPUTE` (SAS Institue Inc., 2015), which partly covers some fractional imputation methods.

Fractional imputation is yet to be widely used in practice other than survey sampling possibly because it is relatively new and there is more complexity involved in implementing variance estimation compared to multiple imputation. However, fractional imputation has its own advantages. For instance, in the case of using a method-of-moment estimator, fractional imputation provides consistent variance estimation while the multiple imputation variance estimator is inconsistent (Yang and Kim, 2016).

Imputing multivariate missing data is challenging for both multiple imputation and fractional imputation. In practice, a full conditional specification of the joint model can be used to fill in several missing items. One popular method is multiple imputation using chained equations, also named **mice** (van Buuren and Groothuis-Oudshoorn, 2011) as in the name of its R package. This multiple imputation procedure by chained equation (MICE) involves a variable-by-variable approach using chained equations, whereby the imputation model for each missing item is separately specified with the other items as predictors. A linear regression model or predictive mean matching is used for continuous variables, and a logistic regression is used for categorical variables. Once all the conditional distributions are specified, the multiple imputation procedure is repeated until convergence.

However, full conditional specification is subject to model mis-specifications and model compatibility problems (Chen, 2010), and the parameter estimation procedure is sometimes cumbersome. As an alternative, we employ the fractional hot deck imputation (FHDI) proposed by Im et al. (2015), which is a nonparametric imputation approach using a two-phase sampling idea. To apply FHDI in multivariate missing data, Im et al. (2015) first created imputation cells to match donors and recipients in a nonparametric way, where units with complete data serve as donors and units with at least one missing item serve as recipients. Once the imputation cells are created, multiple donors are assigned to each recipient with the probability proportional to fractional weights, which can be understood as the conditional probability of obtaining the imputed values given the partial observation. After that, the observed values of each donor are jointly imputed to fill the missing parts of the recipient.

The complete R package **FHDI** and entire source codes are available in Im et al. (2018), and can be installed with R 3.4.0 or higher. The package basically includes three main functions with separate purposes. The function `FHDI_CellMake()` converts continuous data into categorical data which can be used as imputation cells. The function `FHDI_CellProb()` is provided to estimate the imputation cell probability using a version of the expectation maximization (EM) algorithm. This function can be used to get the maximum likelihood estimates of the cell probabilities from multivariate, incomplete, categorical data under the missing at random assumption. The function `FHDI_Driver()` performs fractional hot deck imputation, and also provides a set of replicated fractional weights for variance estimation.

FHDI

Fractional hot deck imputation (FHDI), proposed by Kim and Fuller (2004), replaces each missing value with a set of imputed values. Those imputed values are selected at random from values of the donors in the same imputation cell, with the cells constructed to achieve within-cell data homogeneity. Fractional weights are assigned to each imputed value to preserve the original data structure, and a singly imputed data set is obtained as the output of the FHDI.

Im et al. (2015) extended Kim and Fuller (2004)'s idea in two ways. First, the new FHDI does not require imputation cells to be made in advance. Imputation cells are determined as a by-product of the imputation procedure, and are generally created to preserve the most of the correlations among survey items. Second, the new FHDI method is now applied to multivariate missing data with arbitrary missing patterns.

The FHDI of Im et al. (2015) can be understood as an imputation method using two-phase sampling for stratification. In phase one, the imputation cells are determined so that the survey values are homogeneous within cells and each missing unit has at least two possible donors within each imputation cell. The cell probabilities are estimated using the EM by weighting method (Ibrahim, 1990). Once imputation cells are fixed, then the fully efficient fractional imputation (FEFI), named by Fuller and Kim (2005), is implemented by replacing each missing value with all observed values within the imputation cell. The FEFI fractional weights are obtained during this FEFI procedure. However, in general, the size of imputed data from the FEFI procedure can be too large to handle for statistical analysis. To avoid this size issue, we may select $M(\geq 2)$ donors on each missing unit instead of taking all possible donors. In phase two, a set of donors are assigned to each recipient with the probability proportional to their FEFI fractional weights. A vector of missing values is jointly imputed with the observed values of assigned donors.

Basic setup

For a description of the FHDI procedure, suppose that we have a finite population of size N, indexed by $U = \{1, 2, \ldots, N\}$, with two continuous variables y_1 and y_2. Let z_1 and z_2 be discretized values of y_1 and y_2, respectively. Assume that z_1 takes values in $\{1, \ldots, G\}$ and z_2 takes values in $\{1, \ldots, H\}$. Let δ_p $(p = 1, 2)$, a response indicator function of y_p, take a value of one if y_p is observed and zero otherwise. Note that if y_p has a missing value, then z_p is also missing.

The finite population U can be subdivided into $G \times H$ cells based on z_1 and z_2, and we assume a cell mean model on the cells such that

$$y \mid (z_1 = g, z_2 = h) \sim (\mu_{gh}, \Sigma_{gh}), \quad g = 1, \ldots, G,\ h = 1, \ldots, H, \tag{1}$$

where $y = (y_1, y_2)$, $\mu_{gh} = (\mu_{1,gh}, \mu_{2,gh})$ is a vector of cell means and Σ_{gh} is the variance-covariance matrix of y in cell (gh).

Let y_{obs} and y_{mis} be the observed and missing part of y, respectively. We assume that the data are missing at random (MAR) in the sense that

$$P(\delta \mid y) = P(\delta \mid y_{obs}), \tag{2}$$

where $\delta = (\delta_1, \delta_2)$. The MAR condition (2) implies that the imputation model (1) also holds for the respondents.

Let A be the index set of the sample elements selected from the finite population U. Let A_R be the index set of the respondents who answered both items y_1 and y_2, that is, $A_R = \{j \in A; \delta_{1j}\delta_{2j} = 1\}$. Similarly, define A_M to be the set of the nonrespondents who have at least one missing value. That is, $A_M = \{j \in A; \delta_{1j}\delta_{2j} = 0\}$. Denote n_R and n_M as the size of A_R and A_M, respectively. We assume that A_R is non-empty and there exists enough donors in A_R, that is,

$$n_R \geq n_R^* \tag{3}$$

for some n_R^*. The value of n_R^* is determined by the size of the imputation cells.

Imputation

The FHDI procedure in Im et al. (2015) consists of the following four steps: (i) Cell construction by discretization, (ii) Estimating cell probabilities, (iii) Constructing FEFI fractional weights, and (iv) Imputation. A detailed step-by-step description is provided below.

(Step 1): Cell construction by discretization

We wish to construct imputation cells satisfying (1). The imputation cell variable z can be given in advance, or can be obtained using the estimated sample quantiles. To discuss the latter case, let $\{a_1, \ldots, a_G\}$ be a set of cumulative proportions such that $0 = a_0 < a_1 < \cdots < a_{G-1} < a_G = 1$. We can choose a_i so that each cell contains an equal number of respondents. Let

$$\hat{F}(t) = \frac{\sum_{i \in A} \delta_{1i} w_i I(y_{1i} \leq t)}{\sum_{i \in A} \delta_{1i} w_i} \tag{4}$$

be the estimated distribution function for y_1, where w_i is the sampling weight of unit i; $I(S)$ is an indicator function that takes value of one if S is true and zero otherwise; and $\hat{q}(a_k)$ be the estimated sample quantile corresponding to a_k, defined by $\hat{q}(a_k) = \min\{t; \hat{F}(t) \geq a_k\}$. Once we have the estimated sample quantiles, we can construct z_{1i} from y_{1i}. For instance, $z_{1i} = g$ if $\hat{q}(a_{g-1}) < y_{1i} \leq \hat{q}(a_g)$. If y_{1i} is missing, then z_{1i} has a 'NA' value. Similarly, we can construct z_2 with the range from 1 to H.

From the realized values of z_{1i} and z_{2i} in the sample, we can construct two sets of observed patterns of (z_1, z_2) for A_R and A_M. Let V_R be the set of all observed combinations of z_1 and z_2 in A_R. A size of V_R is $G \times H$ at maximum, but it can be smaller in the realized samples. Similarly we obtain V_M based on the observed parts of nonrespondents. For example, we may have $V_M = \{(\text{NA}, \text{NA}), (\text{NA}, z_2 = 1), (\text{NA}, z_2 = 2), (z_1 = 1, \text{NA}), (z_1 = 2, \text{NA})\}$ in the case of two binary outcomes.

For the proposed FHDI, we need at least two donors for each recipient to capture the variability from imputation. However, an initial discretization may not give enough donors for some recipients. In this case, we apply a cell collapsing procedure to have at least two donors to each recipient. During the cell collapsing, the dimension of the cell variable z can be adjusted to have larger samples within the cells. However, each cell variable z should have at least two distinct values marginally for its validity in the construction of imputation cell. The cell collapsing procedure is designed to be stopped if any cell variable has a single observation. In the left panel of Table 1, we only have one possible donor for recipients with $(z_1 = NA, z_2 = 1)$. In the right panel of Table 2, two cells, $(z_1 = 1, z_2 = 1)$ and $(z_1 = 1, z_2 = 2)$, are merged into a single cell to guarantee a larger size of donors to the recipients. As the result of cell collapsing, recipients who have observed values in z_2 with $z_2 = 1$ or $z_2 = 2$ share the same donors if they have $z_1 = 1$. Since the cell variables z_2 have two distinct values given $z_1 = 2$, the cell collapsing result does not contradict the overall size assumption of $H \geq 2$.

$z_1\backslash z_2$	1	2
1	0	3
2	1	4

$z_1\backslash z_2$	1	2
1	3	
2	1	4

Table 1: An illustrative example for the number of donors in cell with $G = 2$ and $H = 2$. Left panel: initial imputation cell; right panel: final cell subdivision after cell collapsing.

(Step 2): Estimating of cell probabilities

Once the imputation cells are finalized from the above discretization, we need to estimate the cell probabilities defined by

$$\pi_{gh} = P(z_1 = g, z_2 = h), \quad g = 1, \ldots, G;\ h = 1, \ldots, H.$$

The initial cell probabilities are obtained using only the respondents in A_R. These initial cell probabilities are updated using the following EM method, modified from the EM by weighting (Ibrahim, 1990):

E-step: Let z_{obs} and z_{mis} be observed and missing part of z, respectively. Then, the conditional probability of $z_{mis} = b^*$ given $z_{obs} = a$, denoted by $\hat{\pi}_{b^*|a}$, is computed using the current t-th estimate of the joint probability, where a is the observed value in z_{obs}, b^* is one possible value for z^*_{mis}, and

$$\hat{\pi}^{(t)}_{b^*|a} = \frac{\hat{P}^{(t)}(z_{i,obs} = a, z_{i,mis} = b^*)}{\sum_b \hat{P}^{(t)}(z_{i,obs} = a, z_{i,mis} = b)}. \tag{5}$$

M-step: Updates the joint probability of a particular combination $z^* = (z_{obs} = a, z^*_{mis} = b^*)$ by

$$\hat{P}^{(t+1)}(z^*) = \left(\sum_{i \in A}^{n} w_i\right)^{-1} \sum_{i \in A}^{n} w_i \hat{\pi}^{(t)}_{b^*|a} I(z_{i,obs} = a). \tag{6}$$

(Step 3): Constructing FEFI fractional weights

The key point of the approach in Im et al. (2015) is to approximate the FEFI by the FHDI method with a smaller size of donors. To achieve this goal, we first need to compute the FEFI fractional weights for all possible donors assigned to each recipient. Let w^*_{ij} be the j-th fractional weights for the recipient i corresponding to donor j given by

$$w^*_{ij} = \hat{\pi}_{z^*_{i,mis}|z_{i,obs}} \frac{w_j I\{(z_{i,obs}, z^*_{i,mis(i)}) = (z_{j,obs(i)}, z_{j,mis(i)})\}}{\sum_{k \in A_R} w_k I\{(z_{i,obs}, z^*_{i,mis(i)}) = (z_{k,obs(i)}, z_{k,mis(i)})\}}, \tag{7}$$

where $z^*_{i,mis}$ is an imputed value for the missing part of recipient i and $(z_{k,obs(i)}, z_{k,mis(i)})$ denotes the values of unit k corresponding to the observed and missing part of recipient i in the sample imputation cell. Here, the FEFI fractional weights are constructed using the cell conditional probability of $z_{i,mis}$ in the missing part given the observed part $z_{i,obs}$. Note that the sum of w^*_{ij} over all j is equal to one by construction.

(Step 4): Imputation

In FEFI, we employed all respondents as donors to each recipient in the same cell, and then assign the FEFI fractional weights to each donor. However, this FEFI may not be attractive in practice due to its huge size. Instead of using all the respondents, we can select just M donors among the FEFI donors with the selection probability proportional to FEFI fractional weights and then assign equal fractional weights. As for donor selection, we used a tailored systematic sampling method given below:

(a) Sort all FEFI donors in y values by the half-ascending half-descending order. That is, for example, $\{1, 2, \ldots, 8\}$ is sorted as follows: 1,3,5,7,8,6,4,2.

(b) Let $k \in A_{Rd}$, where A_{Rd} is a set of the FEFI donors, be the FEFI donors after the sorting algorithm in (a). Let (L_k, U_k) be the interval for the following systematic sampling scheme:

  (b1) $L_1 = 0$. Set $k = 1$.

  (b2) For current k, $U_k = L_k + w^*_{ik}$.

  (b3) Set $k = k + 1$ and $L_k = U_{k-1}$ then go to Step (b2) until $k = n_{Rd}$, where n_{Rd} is a size of A_{Rd}.

(c) Let A_{Mr} be a subset of A_M who has the same values in the observed part and n_{Mr} be the size of A_{Mr}. Let RN be a random number generated from a uniform distribution $U(0, 1)$. For each $l \in A_{Mr}$, we select M donors as follows: for $j = 1, \ldots, M$, if

$$L_k \leq \frac{RN + (l-1)}{n_{Mr}} + (j-1) \leq U_k$$

for some k, then a donor k is selected as the j-th donor for recipient l.

This tailored systematic sampling is designed to select the FEFI donors efficiently within imputation cells. If n_{Rd} is less than M, we select all donors and then assign the FEFI fractional weights instead of assigning equal weights M^{-1}.

Analysis

After imputation, we obtain the imputed data in the size of $n_R + n_M \times M$. On the imputed data, we can conduct statistical analysis such as mean estimation, regression analysis, and so on. The FHDI mean estimator is

$$\bar{y} = \frac{\sum_{i\in A}\sum_{j=1}^{M} w_i w_{ij}^* y_{ij}^*}{\sum_{i\in A} w_i}, \tag{8}$$

where y_{ij}^* and w_{ij}^* are the imputed values and the fractional weights for unit $i \in A_M$, but $y_{ij}^* = y_i$ for all j and $w_{ij}^* = 1$ for $i = j$ and $w_{ij}^* = 0$ for all $i \neq j$ if unit $i \in A_R$.

Similarly, the regression coefficient for the regression of y_1 given y_2 can be written as a classical weighted regression in the form,

$$\hat{\beta} = (X^{'}WX)^{-1}X^{'}Wy_1^*, \tag{9}$$

where X is a design matrix including a vector of $y_{2,ij}^*$, and W is a weighting matrix whose diagonal elements are $w_i w_{ij}^*$ and non-diagonal elements are all zeros.

The replication method is considered for variance estimation of the FHDI estimator. For L replicates, the replication variance estimator for the FEFI estimator is

$$\hat{\theta}_{FHDI} = \sum_{k=1}^{L} c_k \left(\hat{\theta}_{FHDI}^{(k)} - \hat{\theta}_{FHDI}\right)^2, \tag{10}$$

where c_k is a replicate factor associated with $\hat{\theta}_{FHDI}^{(k)}$ and $\hat{\theta}_{FHDI}^{(k)}$ is the the k-th replicate estimate obtained using the k-th fractional weights replicate denoted by $w_i^{(k)} \times w_{ij}^{*(k)}$. The current version of the **FHDI** package provides a set of replication fractional weights using a jackknife method. See Im et al. (2015) for details in computation of the replication fractional weights.

Implementation of FHDI

In the **FHDI** package, we have three main functions: (i) `FHDI_CellMake`, (ii) `FHDI_CellProb`, and (iii) `FHDI_Driver`. The function `FHDI_CellMake()` is used to create the imputation cell variable z. The EM algorithm introduced in Section 2.2 is built into the function `FHDI_CellProb()`. The main function `FHDI_Driver()` conducts imputation and variance estimation including cell construction and estimating cell probabilities. We used simulated data to describe these components of the package **FHDI**. All results in this section are obtained from a Microsoft Windows 64 bit operation system. The FEFI results should be the same without reference to the underlying operating system, while the FHDI results using other platforms (e.g., Linux) may be slightly different, as FHDI selects donors using a standard random number library which is generally platform-dependent.

DATA

We have $n = 100$ sample observations for the multivariate data vector $y_i = (y_{1i}, y_{2i}, y_{3i}, y_{4i})$, $i = 1, \ldots, n$, generated from

$$\begin{aligned} Y_1 &= 1 + e_1, \\ Y_2 &= 2 + \rho e_1 + \sqrt{1-\rho^2}e_2, \\ Y_3 &= Y_1 + e_3, \\ Y_4 &= -1 + 0.5Y_3 + e_4. \end{aligned}$$

We set $\rho = 0.5$; e_1 and e_2 are generated from a standard normal distribution; e_3 is generated from a standard exponential distribution; and e_4 is generated from a normal distribution $N(0, 3/2)$.

Response indicators are generated from a Bernoulli distribution with different p_k,

$$\delta_k \sim B(p_k),$$

where $(p_1, p_2, p_3, p_4) = (0.6, 0.7, 0.8, 0.9)$. Although the response indicators are generated based on the missing completely at random (MCAR) assumption for simplicity, the FHDI method also holds for other response models based on MAR.

Based on the outcome models and the response models above, an example data are generated using the following R code:

```
n = 100
set.seed(1345)
rho = 0.5
e1 = rnorm(n, 0, 1)
e2 = rnorm(n, 0, 1)
e3 = rgamma(n, 1, 1)
e4 = rnorm(n, 0, sd = sqrt(3/2))

y1 = 1 + e1
y2 = 2 + rho * e1 + sqrt(1 - rho^2) * e2
y3 = y1 + e3
y4 = -1 + 0.5 * y3 + e4

r1 = rbinom(n, 1, p = 0.6)
r2 = rbinom(n, 1, p = 0.7)
r3 = rbinom(n, 1, p = 0.8)
r4 = rbinom(n, 1, p = 0.9)

y1[r1 == 0] = NA
y2[r2 == 0] = NA
y3[r3 == 0] = NA
y4[r4 == 0] = NA
```

Imputation cell

Using the R function `summary()`, we see missing values in all four variables; realized response rates are 0.58, 0.66, 0.82, and 0.89, respectively; and sample means for complete cases are 0.98, 1.93, 1.80, and -0.01, respectively.

```
> daty = cbind(y1, y2, y3, y4)  # data
> datr = cbind(r1, r2, r3, r4)  # response (0:mising cell;  1: observed cell)
> summary(daty)
      y1                 y2                 y3                 y4
Min.   :-1.6701   Min.   :0.02766   Min.   :-1.4818   Min.   :-2.920292
1st Qu.: 0.4369   1st Qu.:1.03796   1st Qu.: 0.9339   1st Qu.:-0.781067
Median : 0.8550   Median :1.79693   Median : 1.7246   Median :-0.121467
Mean   : 0.9821   Mean   :1.93066   Mean   : 1.7955   Mean   :-0.006254
3rd Qu.: 1.6171   3rd Qu.:2.71396   3rd Qu.: 2.5172   3rd Qu.: 0.787863
Max.   : 3.1312   Max.   :5.07103   Max.   : 5.3347   Max.   : 4.351372
NA's   :42        NA's   :34        NA's   :18        NA's   :11
```

We now use the function `FHDI_CellMake()` to convert continuous y variables into discretized variables z. A vector of initial cell dimension k should be given as input information. If the input value k is a single integer, the same number of category is applied to all variables for initial discretization. The current version allows up to 35 distinct categories for each variable. When the variables are specified as the unordered categorical type through the option, the algorithm excludes those variables from the discretization procedure. The details are illustrated in the a ppendix along with the application of mixed data types.

Sample weight and ID are optional for input information. Default values are 1 through n for the ID while 1.0 for the sample weight. Another option is `s_op_merge` which controls randomness in the cell collapsing procedure. There are two possible values for `s_op_merge`: `"fixed"` as default and `"rand"` as optional. During the cell collapsing, randomness occur if there exists multiple adjacent cells for a fixed cell, which is required to be merged into another cell. If `s_op_merge` is set with `"rand"`, then the matrix of discretized values can be different even for the same incomplete data. The last option is `"categorical"` used to denote the data type which has a value 1 if the input variable is unordered and 0 otherwise. If the input vector for the option `"categorical"` is not specified by the users, the algorithm treats all variables as continuous or ordered categorical types.

The main outputs of the function `FHDI_CellMake()` are (a) incomplete data matrix attached with ID and sample weight named by “`data`", (b) matrix for imputation cell variables named by “`cell`", (c) cell pattern matrix for respondents named by “`cell.resp`", and (d) cell pattern matrix for nonrespondents named by “`cell.non.resp`". When missing values are recorded with specific numeric values other than `NA`, the response indicator matrix (i.e., “`datr`") should be inserted with the original data matrix. The indicator matrix “`datr`" has the same dimension as “daty" contains 0 for missing cell locations and 1 otherwise. For instance, the first six rows of “`datr`" of the “`daty`" given above will look like:

```
y1  y2  y3  y4
 1   1   0   1
 0   1   1   1
 1   1   1   0
 0   1   0   1
 1   1   1   1
 1   1   1   1
```

As long as "`daty`" contains `NA` at the missing cell locations, automatic detection of missing cell locations takes place. When one wants to define more missing cell locations, one can override the automatic detection by separately defining "datr" and including it in the function argument, for example, '`FHDI_CellMake(daty,datr,k=3)`'. We set $k = 3$ and do not give additional input information on weights and ID, then the first output is shown as

```
> cdaty = FHDI_CellMake(daty, k = 3)
> names(cdaty)
[1] "data"           "cell"           "cell.resp"      "cell.non.resp"
[5] "w"              "s_op_merge"
> head(cdaty$data)
     ID WT         y1       y2        y3           y4
[1,]  1  1 1.47963286 2.150860        NA  1.894211796
[2,]  2  1         NA 1.141496 1.6025296 -1.036946859
[3,]  3  1 0.70870936 1.885673 1.2506894           NA
[4,]  4  1         NA 2.753840        NA  1.211049509
[5,]  5  1 0.86273572 2.425549 1.8875492 -0.539284732
[6,]  6  1 0.03460025 1.740481 0.4909525  0.007130484
```

The observed values are converted to ordered categorical values from 1 to k. Here, missing values are presented with a common missing value "`0`". During the discretization procedure, the initial dimension of k can be down to have larger donors within imputation cells. In our example, the default value $k = 3$ was kept for all variables.

```
> head(cdaty$cell)
     y1 y2 y3 y4
[1,]  3  2  0  3
[2,]  0  1  1  1
[3,]  2  2  2  0
[4,]  0  2  0  3
[5,]  2  3  2  2
[6,]  1  2  1  2

> apply(cdaty$cell, 2, table)
  y1 y2 y3 y4
0 42 34 18 11
1 26 18 27 30
2 13 36 32 28
3 19 12 23 31
```

From the output component named by "`cell`", we can find out whether the cells are merged during the discretization process. For instance, we have 34 missing values in y_2, and this indicates that 22 $(= 66/3)$ observations are evenly distributed for each category based on the estimated density function. However, the finalized frequencies for three categories are 18, 36, and 12. This means that some initial cells are merged to adjacent cells to obtain enough donors for recipients.

It is important to specify the cell patterns to match the respondents and the nonrespondents. Thus, the cell pattern matrix for respondents and nonrespondents, named by "`cell.resp`" and "`cell.non.resp`", are produced as the outputs of the function `FHDI_CellMake()`. Note that it is possible to have $3^4 = 81$ distinct vectors theoretically, but we only have 10 unique patterns in our toy example.

```
> cdaty$cell.resp
     y1 y2 y3 y4
[1,]  1  1  1  1
[2,]  1  1  2  3
[3,]  1  2  1  2
[4,]  1  2  2  1
```

```
[5,]  2  2  2  3
[6,]  2  3  2  2
[7,]  3  1  3  3
[8,]  3  2  3  2
[9,]  3  2  3  3
[10,]  3  3  3  1
```

Similarly, a set of unique cell patterns for nonrespondents are reported at the fourth output. We have 47 patterns for nonrespondents in this example. Some patterns are presented below

```
> head(cdaty$cell.non.resp)
     y1 y2 y3 y4
[1,]  0  0  0  2
[2,]  0  0  0  3
[3,]  0  0  1  1
[4,]  0  0  1  2
[5,]  0  0  2  1
[6,]  0  0  2  2
```

Note that it is possible to have all zero values as a cell pattern, because we allow to have missing values for all variables. Under the MAR assumption, if there are no additional auxiliary variables, then the cases with missing values in all items can be safely removed from the original data set.

Cell probability estimation

We now estimate the cell probabilities based on the second output obtained using `FHDI_CellMake()`. Since the function `FHDI_CellProb()` is based on the EM algorithm designed to compute cell probabilities for multivariate categorical data, this function can be separately used when we are only interested in obtaining the maximum likelihood estimates for the cell probabilities.

```
> datz = cdaty$cell
> jcp = FHDI_CellProb(datz)
> jcp$cellpr
1111       1123       1212       1221       2223       2322
0.18110421 0.05474648 0.12693514 0.07786676 0.17388579 0.08263912
3133       3232       3233       3331
0.02175015 0.10356376 0.08871434 0.08879425
> sum(jcp$cellpr)
[1] 1
```

Note that the joint cell probabilities are estimated only from observed patterns, not from all possible values. Overall sum of the joint cell probabilities should be equal to one. All discretized values are presented by a single values in the order of data columns. For example, a value of “1111" denotes a vector $(z_1 = 1, z_2 = 1, z_3 = 1, z_4 = 1)$. If the number of categories in a variable is larger than 10, the categories are label with 26 alphabet letters (a-z). If we have a value ‘b2c’, then it denotes the z vector $(11, 2, 12)$.

Fractional hot deck imputation

We can use the function `FHDI_Driver()` without searching for a suitable imputation cell matrix in advance. In short, `FHDI_Driver()` automatically performs `FHDI_CellMake()` and `FHDI_CellProb()` and then proceeds toward the imputation and/or variance estimation. The main input information is the matrix of original variables, matrix for response indicators, and the imputation method (`"FEFI"`, `"FHDI"`). In the case of ‘`s_op_imputation="FHDI"`’, the imputation size M should be given as an input value. For instance, when $M = 5$, FHDI will randomly select 5 donors from all possible donors. In the case of `"FEFI"`, the imputation is conducted to assign all possible values to each recipient with the FEFI fractional weights.

The output consists of four main parts except for input information: (i) imputation results with fractional weights named by “`fimp.data`", (ii) an imputed data in format of single imputation result named by “`simp.data`", (iii) the FHDI mean estimates with estimated standard error named by "`imp.mean`", and (iv) replication fractional weights for variance estimation named by "`rep.weight`". If variance estimation option is given by ‘`i_op_variance=0`’, then third output and fourth output are not produced as the main output. The default is ‘`i_op_variance=1`’.

The part of the fractionally imputed data are given below with newly attached variables FID and FWT, where FID denotes donors' local serial index and FWT denotes fractional weights assigned to imputed values:

```
> FEFI = FHDI_Driver(daty, s_op_imputation = "FEFI", i_op_variance = 1, k = 3)
> names(FEFI)
[1] "fimp.data"       "simp.data"       "imp.mean"
[4] "rep.weight"      "M"               "s_op_imputation"
[7] "i_option_merge"

> FEFI$fimp.data[1:20,]
      ID FID WT       FWT          y1       y2       y3         y4
 [1,]  1   1  1 0.5000000  1.47963286 2.150860 2.881646  1.8942118
 [2,]  1   2  1 0.5000000  1.47963286 2.150860 2.493438  1.8942118
 [3,]  2   1  1 0.2000000 -0.09087472 1.141496 1.602530 -1.0369469
 [4,]  2   2  1 0.2000000 -1.67006193 1.141496 1.602530 -1.0369469
 [5,]  2   3  1 0.2000000 -0.39302750 1.141496 1.602530 -1.0369469
 [6,]  2   4  1 0.2000000  0.97612864 1.141496 1.602530 -1.0369469
 [7,]  2   5  1 0.2000000  0.21467221 1.141496 1.602530 -1.0369469
 [8,]  3   1  1 0.1666667  0.70870936 1.885673 1.250689  0.7770526
 [9,]  3   2  1 0.1666667  0.70870936 1.885673 1.250689  1.2839115
[10,]  3   3  1 0.1666667  0.70870936 1.885673 1.250689  0.6309413
[11,]  3   4  1 0.1666667  0.70870936 1.885673 1.250689  0.3232018
[12,]  3   5  1 0.1666667  0.70870936 1.885673 1.250689  0.5848844
[13,]  3   6  1 0.1666667  0.70870936 1.885673 1.250689  1.0342970
[14,]  4   1  1 0.1103616  1.22307261 2.753840 1.923865  1.2110495
[15,]  4   2  1 0.1689153  2.29021618 2.753840 2.881646  1.2110495
[16,]  4   3  1 0.1103616  0.86825894 2.753840 1.086562  1.2110495
[17,]  4   4  1 0.1103616  2.16515160 2.753840 2.311461  1.2110495
[18,]  4   5  1 0.1103616  0.79827971 2.753840 3.040016  1.2110495
[19,]  4   6  1 0.1689153  2.01819696 2.753840 2.493438  1.2110495
[20,]  4   7  1 0.1103616  0.73228949 2.753840 3.422369  1.2110495
```

The singly imputed data has the same size as the original incomplete data in which missing values are filled with a single value, essentially the mean of fractionally imputed values. For example, in the first sample presented at 'daty[1,]' below, we have a missing value for y_3. The imputed values are 2.881646 and 2.493438 presented above, and the weighted mean of two values 2.6875422, considering sample weights and fractional weights, replaces the missing value. This second output can be used as the result for single imputation. However, its uses should be controlled under the limitations of single imputation. The second output is partially presented below

```
> daty[1,]
      y1       y2       y3       y4
1.479633 2.150860       NA 1.894212

> head(FEFI$simp.data)
              y1       y2        y3           y4
[1,]  1.47963286 2.150860 2.6875422  1.894211796
[2,] -0.19263266 1.141496 1.6025296 -1.036946859
[3,]  0.70870936 1.885673 1.2506894  0.772381422
[4,]  1.44470586 2.753840 2.3372705  1.211049509
[5,]  0.86273572 2.425549 1.8875492 -0.539284732
[6,]  0.03460025 1.740481 0.4909525  0.007130484
```

The component "imp.mean" shows the FEFI mean estimates with the standard errors. The first row presents the FEFI mean estimates and the second row presents the standard error of the FEFI mean estimates. In our example, the FEFI mean estimates (standard errors) for variables are approximately $0.90(0.128)$, $1.87(0.121)$, $1.82(0.137)$, and $-0.03(0.130)$, respectively.

```
> FEFI$imp.mean
          [,1]      [,2]      [,3]        [,4]
[1,] 0.9049227 1.8668846 1.8188381 -0.03193875
[2,] 0.1275504 0.1206944 0.1369989  0.12958845
```

The standard errors are computed using the variance formula in (10) with the replicated fractional weights reported in the fourth output. In a random sample, the associate factor c_k has the same value for all replicates. For instance, $c_k = (n-1)/n$ for the jackknife variance estimation. Replication fractional weights for the imputed data are given below:

```
> dim(FEFI$rep.weight)
[1] 330 100
> FEFI$rep.weight[1:13, 1:6]
           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
 [1,] 0.0000000 0.5050505 0.5050505 0.5050505 0.5050505 0.5050505
 [2,] 0.0000000 0.5050505 0.5050505 0.5050505 0.5050505 0.5050505
 [3,] 0.2020202 0.0000000 0.2020202 0.2020202 0.2020202 0.2020202
 [4,] 0.2020202 0.0000000 0.2020202 0.2020202 0.2020202 0.2020202
 [5,] 0.2020202 0.0000000 0.2020202 0.2020202 0.2020202 0.2020202
 [6,] 0.2020202 0.0000000 0.2020202 0.2020202 0.2020202 0.2020202
 [7,] 0.2020202 0.0000000 0.2020202 0.2020202 0.2020202 0.2020202
 [8,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
 [9,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
[10,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
[11,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
[12,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
[13,] 0.1683502 0.1683502 0.0000000 0.1683502 0.1683502 0.1683502
```

Because a jackknife method is used for variance estimation, a matrix of the replication fractional weights has n_I rows and n columns, where $n_I(> n)$ denotes the size of the imputed data. By its construction, the column sums of the replication fractional weights should be the sum of unit weights, that is, $\sum_{i=1}^{n} w_i^{(k)} = \sum_{i=1}^{n} w_i$ holds for all k.

Instead of the FEFI, we can perform a general FHDI with $M(\geq 2)$. If the imputation method is chosen to be `"FHDI"` and an imputation size M is given, then M donors are assigned to each recipients. If the number of donors for a recipient is smaller than M, all donors are selected and assigned the FEFI fractional weights.

Note that in the recipient with ID=3 (see '`FEFI$fimp.data[1:20,]`' results shown above), there are six donors for FEFI but there are five donors ($M = 5$) for FHDI (see '`FHDI$fimp.data[1:14,]`' results shown below). Five donors are randomly selected among all possible six donors with the probability proportional to the FEFI fractional weights. This indicates that we have additional randomness due to donor selection in the FHDI method. As shown in '`FHDI$imp.mean`' results below, the FHDI mean estimates (standard errors) are approximately 0.90(0.129), 1.87(0.121), 1.82(0.137), and $-0.04(0.131)$, respectively. Compared to the FEFI estimates, both point and variance estimates are similar to each other. This indicates that the FHDI estimator well approximates the FEFI estimator with smaller size of the imputed data. The results (which can vary slightly between operating systems) are presented below:

```
> FHDI = FHDI_Driver(daty, s_op_imputation="FHDI", M=5, i_op_variance=1, k=3)
> FHDI$fimp.data[1:14,]
      ID FID WT FWT          y1       y2       y3         y4
 [1,]  1   1  1 0.5  1.47963286 2.150860 2.881646  1.8942118
 [2,]  1   2  1 0.5  1.47963286 2.150860 2.493438  1.8942118
 [3,]  2   1  1 0.2 -0.09087472 1.141496 1.602530 -1.0369469
 [4,]  2   2  1 0.2 -1.67006193 1.141496 1.602530 -1.0369469
 [5,]  2   3  1 0.2 -0.39302750 1.141496 1.602530 -1.0369469
 [6,]  2   4  1 0.2  0.97612864 1.141496 1.602530 -1.0369469
 [7,]  2   5  1 0.2  0.21467221 1.141496 1.602530 -1.0369469
 [8,]  3   1  1 0.2  0.70870936 1.885673 1.250689  0.7770526
 [9,]  3   2  1 0.2  0.70870936 1.885673 1.250689  1.2839115
[10,]  3   3  1 0.2  0.70870936 1.885673 1.250689  0.6309413
[11,]  3   4  1 0.2  0.70870936 1.885673 1.250689  0.3232018
[12,]  3   5  1 0.2  0.70870936 1.885673 1.250689  1.0342970
[13,]  4   1  1 0.2  1.22307261 2.753840 1.923865  1.2110495
[14,]  4   2  1 0.2  2.29021618 2.753840 2.881646  1.2110495

> FHDI$imp.mean
[,1]     [,2]      [,3]       [,4]
[1,] 0.9032611 1.865013 1.8204121 -0.03900379
[2,] 0.1291721 0.120509 0.1367363  0.13086824
```

Table 2 presents the standard errors of the three mean estimators. Here, the Naive estimator is just a simple mean estimator computed using only observed values. Since the partially observed values are used in the mean estimation, the two estimators obtained using fractional hot deck imputation produce smaller standard errors compared to the Naive estimator.

Estimator	y_1	y_2	y_3	y_4
Naive	0.135	0.135	0.150	0.138
FEFI	0.128	0.121	0.137	0.130
FHDI	0.129	0.121	0.137	0.131

Table 2: Standard errors of three mean estimators.

It should be noted that the automatically generated "datz" can be replaced with a user-defined one. If the imputation cell matrix "datz" is separately obtained from FHDI_CellMake() or provided by the user, it needs to be specified as an option of the function FHDI_Driver(). An example code is given below:

```
> FEFI=FHDI_Driver(daty,datz,s_op_imputation="FEFI",i_op_variance=1,k=3)
```

If survey variables are all categorical, the original data "daty" can be directly used for "datz." In the case of mixed types including continuous and/or ordered and unordered categorical data, we need two steps to obtain the imputation cell matrix: (i) discretization procedure for the continuous parts, and (ii) combining procedure for the converted data and the unordered categorical data. If an error message is presented due to insufficient of the donors from the function FHDI_CellMake(), then the cell collapsing procedure has to be engaged manually for the categorical part. We illustrate an example in Appendix when and how we manually merge categories in practice.

Regression analysis

We now consider a regression analysis using the imputed data. The imputed data can be treated as complete data with assigned fractional weights. Thus, the regression coefficient estimates of y_1 given y_2 can be directly obtained using the lm() function. Compared to using only observed samples, the fractional weights are given as input information. The estimates are also obtained using a classical weighted regression estimator presented in Section 2.2. The R codes for obtaining the regression coefficient estimates for the three estimators are given below:

```
> reg.naive = lm(y1 ~ y2, data = as.data.frame(daty))
> reg.naive$coeff
(Intercept)          y2
-0.07425917  0.58798028
> summary(reg.naive)$coeff[,2]
(Intercept)          y2
  0.3050262   0.1424451

> i.daty = as.data.frame(FEFI$fimp.data)
> reg.fefi = lm(y1 ~ y2, data = i.daty, weights = FWT)
> reg.fefi$coeff
(Intercept)          y2
 0.03465671  0.46615949

> i.daty2 = as.data.frame(FHDI$fimp.data)
> reg.fhdi = lm(y1 ~ y2, data = i.daty2, weights = FWT)
> reg.fhdi$coeff
(Intercept)          y2
  0.0227328   0.4721299
```

Note that if we directly use lm() on the imputed data, the standard errors are 0.103 and 0.048 for the FEFI estimator and 0.111 and 0.052 for the FHDI estimator, respectively. However, the standard errors using the replication method are nearly 0.251 and 0.094 for both the FEFI estimator and FHDI estimator. Because the variances coming from imputation are not captured with a classical variance estimator of the regression estimator, variance estimation should be conducted using the replication variance estimator formula in (10).

Estimator	Intercept (S.E.)	Slope (S.E.)
True	0	0.5
Naive	-0.074 (0.305)	0.588 (0.142)
FEFI	0.035 (0.103)	0.466 (0.048)
FHDI	0.023 (0.111)	0.472 (0.052)

Table 3: Regression coefficient estimates with standard errors. (S.E.: standard error.)

Table 3 presents the regression coefficient estimates with standard errors for the three estimators. Point estimates of the FEFI and FHDI estimators are much closer to the true values compared to the values of the Naive estimator. Also, two fractional imputation estimators have smaller standard errors than those of the naive estimator. All R codes to obtain these results are given below:

```
> reg.fefi.coef = reg.fefi$coeff
> reg.est = t(apply(FEFI[[4]], 2, function(s) lm(y1 ~ y2, data = i.daty,
    weights = s)$coeff))
> reg.fefi.rep = reg.est - matrix(reg.fefi.coef, n, 2, byrow = TRUE)
> sqrt(apply(reg.fefi.rep^2, 2, sum) * (n - 1)/n)
(Intercept)          y2
 0.25082547  0.09369202
> summary(reg.fefi)$coeff[,2]
(Intercept)          y2
 0.10333038  0.04840829

> reg.fhdi.coef = reg.fhdi$coeff
> reg.est2 = t(apply(FHDI[[4]], 2, function(s) lm(y1 ~ y2, data = i.daty2,
    weights = s)$coeff))
> reg.fhdi.rep = reg.est2 - matrix(reg.fhdi.coef, n, 2, byrow = TRUE)
> sqrt(apply(reg.fhdi.rep^2, 2, sum) * (n - 1)/n)
(Intercept)          y2
 0.25165391  0.09524197
> summary(reg.fhdi)$coeff[,2]
(Intercept)          y2
 0.11149137  0.05230785
```

Conclusion

FHDI is a useful tool for handling item nonresponse. Since FHDI employs a nonparametric estimation of the joint distribution and uses observed values as imputed values, it can be widely accepted in many research and incomplete data analysis. This paper documents the R package **FHDI** to enable R users to perform fractional hot deck imputation as well as fully efficient fractional imputation.

The current version of the package **FHDI** has some limitations. First, the functions within the package are not always applicable for all types of incomplete data. The current imputation algorithms cannot be applied to fill in missing values when there is no fully observed units over all variables. Although the users may assume conditional independence to apply our imputation algorithms, the imputation on the basis of untestable conditional independence assumption may break the original data structure. Second, the unordered categorical data should be handled manually for the insufficient donor cases before the users use `FHDI_Driver()`. However, overcoming this limitation is identical to coming up with a way to implement cell collapsing with no regard to the rationality of categorization. Also, jackknife variance estimation may be not appropriate or efficient for larger or complicated incomplete data. Algorithm-oriented parallel computing methods and substantial issues including variable types and variance estimation options will be targeted in the future update.

Bibliography

H. Y. Chen. Compatibility of conditionally specified models. *Statistics and Probability Letters*, 80:670–677, 2010. [p140]

G. B. Durrant and C. Skinner. Using missing data methods to correct for measurement error in a distribution function. *SM*, 32:25–36, 2006. [p140]

R. E. Fay. Alternative paradigms for the analysis of imputed survey data. *JASA*, 91:490–498, 1996. [p140]

W. A. Fuller and J. K. Kim. Hot deck imputation for the response model. *SM*, 31:139–149, 2005. [p140, 141]

J. Honaker, G. King, and M. Blackwell. Amelia ii: A program for missing data. *Journal of Statistical Software*, 45:1–47, 2011. [p140]

J. G. Ibrahim. Incomplete data in generalized linear models. *JASA*, 85:765–769, 1990. [p141, 143]

J. Im, J. K. Kim, and W. A. Fuller. Two-phase sampling approach to fractional hot deck imputation. In *JSM Proceedings of Survey Research Methodology Section*, pages 1030–1043, Seattle, WA, 2015. asa. [p140, 141, 142, 143, 144]

J. Im, I. H. Cho, and J. K. Kim. *FHDI: Fractional Hot Deck and Fully Efficient Fractional Imputation*, 2018. URL https://CRAN.R-project.org/package=FHDI. [p141]

G. Kalton and L. Kish. Some efficient random imputation methods. *Communications in Statistics-Theory and Methods*, 13:1919–1939, 1984. [p140]

J. K. Kim. Fractional hot deck imputation. *BMK*, 98:119–132, 2011. [p140]

J. K. Kim and W. A. Fuller. Fractional hot deck imputation. *BMK*, 91:559–578, 2004. [p140, 141]

A. Kowarik and M. Templ. Imputation with the R package VIM. *Journal of Statistical Software*, 74:1–16, 2016. [p140]

D. B. Rubin. Inference and missing data. *BMK*, 63:581–592, 1976. [p140]

D. B. Rubin. *Multiple Imputation for Nonresponse in Survey*. John Wiley & Sons, New York, 1987. [p140]

D. B. Rubin. Multiple imputation after 18+ years. *JASA*, 91:473–489, 1996. [p140]

SAS Institue Inc. *SAS/STAT 14.1 User's Guide*. SAS Institue Inc., Cary, NC, 2015. URL http://support.sas.com/documentation/. [p140]

Y. S. Su, A. Gelman, J. Hill, and M. Yajima. Multiple imputation with diagnostics (mice) in r: Opening windows into the black box. *Journal of Statistical Software*, 45:1–31, 2011. [p140]

S. van Buuren and K. Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45:1–67, 2011. [p140]

S. Yang and J. K. Kim. A note on multiple imputation for method of moments estimation. *BMK*, 103: 244–251, 2016. [p140]

Jongho Im
Department of Applied Statistics
Yonsei University
South Korea
ijh38@yonsei.ac.kr

In Ho Cho
Department of Civil, Construction and Environmental Engineering
Iowa State University
United States
icho@iastate.edu

Jae Kwang Kim
Center for Survey Statistics and Methodology
Department of Statistics
Iowa State University
United States
jkim@iastate.edu

Appendix

We here present how R users can handle mixed types, combination of continuous and unordered categorical data, in practice. For illustration purpose, we use the exit poll data collected to predict the 18th South Korean legislative election held in 2008. The exit poll data include data collection channel, the sampled voters' gender, age and candidate. We simply assume the MAR mechanism without any further discussion to guide how the R users handle categorical data or mixed type data in uses of `FHDI_CellMake()`.

Table A.1 gives a summary of the exit poll results in an example district. The data were collected from six different channels, denoted by $1, 2, \ldots, 6$, and the size of exit poll was 2210. Gender was recorded with 1 (Male) and 2 (Female), and age was recorded as a numerical value. There were five parties for the voters' choice, recorded by 1,2,5,6,7. The response rates for the gender, age and candidate were 98%, 98%, and 80%, respectively.

Variable	Type	Size of Support	Observation	Response Rate (%)
Channel	Categorical	6	2210	100
Gender	Categorical	2	2172	98
Age	Numerical	72	2170	98
candidate	Categorical	5	1764	80

Table A.1: A summary of the exit poll results in an example district

For imputation, we first consider the case in which the variables are given as their original types presented in Table A.1. From the input data types, the cell collapsing procedure can be only applied to the variable 'age'. Since three variables, 'channel', 'gender' and 'candidate', require 60 ($= 6 \times 2 \times 5$) imputation cells, the final size will be $60 \times C_{age}$, where C_{age} is the finally discretized cell size of $'age'$. It is easy to fail to have enough donors within each imputation cell due to sparsity problem. When we type the R codes below,

```
> cell.election=FHDI_CellMake(election,k=3,categorical=c(1,1,0,1))
```

we have the following error message:

The current data set does not have enough donors while there is at least one non-collapsible categorical variable!

To make feasible imputation cells, in addition to 'age', we want to also merge parties to reduce the cell dimensions. One possible approach is to un-specify the variable 'candidate' as the unordered categorical values, and the other approach is to merge candidate manually according to the users' background knowledge of the parties. The results for the first approach are presented below:

```
> cell.election = FHDI_CellMake(election, k = 3,categorical = c(1, 1, 0, 0))
> apply(cell.election$cell, 2, table)
$channel
  1   2   3   4   5   6
396 708 508 276 127 195

$gender
  0    1    2
 38 1042 1130

$age
  0   1   2   3
 40 775 672 723

$candidate
  0   1   2   3
446 788 864 112
```

The observed candidates are 1 (788), 2 (864), 5 (73), 6 (27), and 7 (12), and they are merged into three categories 1 (788), 2 (864), and 3 (112). During the cell collapsing, the initial integer values are changed to ordered categorical values and then merged to search out the feasible imputation cells. Although the imputation cells are now well constructed without error message, the cell collapsing results may not be undesirable in the sense that the automatically merged candidates (parties) using the algorithm in the `FHDI_CellMake()` may have the opposite political positions. Note that even if the size of initial categories is given as $k = 3$, the dimension of variables specified as the unordered categorical type is kept to preserve their original sizes.

To avoid irrational cell collapsing result, the second approach is implemented by manually combining the candidates' parties into two groups, 1 (1,6,7) and 2 (2,5), based on their political characteristics. The results with the R codes are given below:

```
> election2 <- election
> election2$candidate[election2$candidate == 5] <- 2
> election2$candidate[election2$candidate == 6] <- 1
> election2$candidate[election2$candidate == 7] <- 1
> cell.election2 = FHDI_CellMake(election2, k = 3, categorical = c(1, 1,
    0, 1))
> apply(cell.election2$cell, 2, table)
$channel
  1   2   3   4   5   6
396 708 508 276 127 195

$gender
  0    1    2
 38 1042 1130

$age
  0   1   2   3
 40 775 672 723

$candidate
  0   1   2
446 827 937
```

As discussed above, there is no errors for both automated and manual cell collapsing procedures. However, the quality of imputation in the automation case can be unsatisfactory due to the inaccuracy of joint distribution approximation. This implies that it is often required to carefully and/or manually handle the unordered categorical variables in practice. Also note that we may want to discretize the variable 'age' with a set of fixed distinct points, for example, (19-29, 30-39, 40-49,50-59, 60+) or (19-39, 40-59, 60+). In this case, the R users also need to manually replace the initial 'age' values to the discretized values in advance of using the function `FHDI_CellMake()`.